Direct digital control of a micro-machine model power system.

Burrows, Peter John

Award date:
1977

Awarding institution:
University of Bath

DIRECT DIGITAL CONTROL OF A MICRO-MACHINE MODEL POWER SYSTEM

by

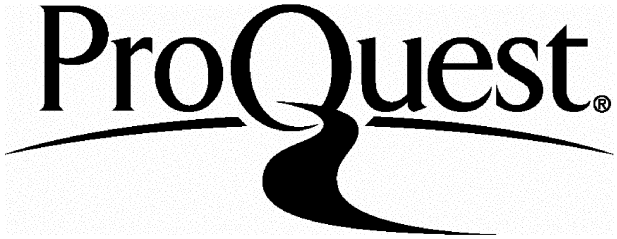
Peter John BURROWS BSc

Thesis submitted for the degree of Doctor of Philosophy of the University of Bath

1977

P J Burrows

October 1977



SUMMARY

This thesis describes the theory, construction and development of a direct digital control scheme for a micro-machine model of an electrical power generating system. The areas of research covered by this thesis fall into four main regions:

1. The design and development of digital transducers to reduce the noise and accuracy problems associated with analogue techniques.

2. The design and development of a fast and flexible real-time interface between the supervisory computer and the controlled system based on the use of dedicated microprocessors.

3. The development of a software real-time operating system and program suite to allow rapid implementation of a required control philosophy for the laboratory system.

4. The practical implementation of sub-optimal excitation control of the synchronous generator by state feedback techniques.

The digital control scheme described has been designed as a flexible base for future work in the implementation of full excitation and governor control of the micromachine system. It is modular in concept to allow rapid application of various forms of modern control philosophy to the laboratory model system.

Various support facilities of wider application have also been developed including a microprocessor to computer interface, a direct-loading cross-assembler and an on-line microprocessor debugging system.

The excitation control implemented in this study involves the addition of state feedback signals to a fast-acting voltage regulator to improve the transient stability limit. The coefficients of the feedback matrix are determined by an off-line parameter optimisation technique to minimise a quadratic performance index of the non-linear system model. This control is shown to effect significant improvements in the system response following the occurrence of a severe disturbance to the system.

CONTENTS

LIST OF PRINCIPAL SYMBOLS

The principal symbols used in this thesis are listed below. Symbols which do not appear in this list will be defined in the text as they are introduced.

A	Plant Matrix
B	Control Matrix
H	Hamiltonian Function
i	Current *
K	Gain (with subscripts denoting parameter)
p	Differential Operator
r	Resistance
s	Laplace Operator
t, T	Time
T_e	Air-Gap Torque
T_m	Motor Torque
u	Control Vector
v	Voltage *
V_f	Field Voltage
V_i	State Feedback Signal
V_r	Reference Voltage to Excitation Regulator
V_t	Terminal Voltage
x	State Vector
x	Reactance *
z	Sensitivity Function
α	Optimisation Parameter
δ	Load Angle

ψ	Flux Linkages *
λ	Eigenvector
ω	Rotor Angular Velocity

* subscripts

a	armature
d	direct axis
q	quadrature axis
fd	field
1d	d-axis damper winding
1q	q-axis damper winding
ad	direct axis mutual
aq	quadrature axis mutual
l	preceding the above, refers to leakage

Other subscripts, etc

T	superscript to denote transpose of a matrix or vector
.	superscript to denote differentiation w.r.t. time
0	subscript to denote steady state value
r	subscript to denote reference
Δ	prefix to denote a deviation about the initial operating point

CHAPTER 1

INTRODUCTION

Rapid technological development in many fields has produced an increasing demand for both quantity and reliability of electrical power. This demand has necessitated developments in plant and system design which, by means of their increased complexity, have demanded a more sophisticated control philosophy to maintain the efficiency and integrity of the system. This need for sophisticated controllers has been met by the introduction of fast, reliable and inexpensive mini- and microcomputers implementing the techniques of modern control theory.

1.1 POWER SYSTEM STABILITY

The increasing demand for electrical power coupled with various environmental pressures has caused the concentration of generation capacity in large efficient power stations in situations remote from the main load centres. The long transmission lines thus required have relatively high impedances causing stability problems particularly when a requirement arises to increase the generating capacity of a particular site. This problem is further aggravated by an increase in size and rating of the individual turbogenerator units causing a reduction in their specific inertia and short-circuit ratio. The shunt capacitance of the long transmission lines has a considerable effect on the

generating system and during periods of light load the generator may have to be run under-excited to absorb the surplus reactive power. It is under these conditions of high load angle that a generator is most likely to lose its synchronism following a transient disturbance and considerable research effort has been expended in increasing the stability limit.

The various limits to the operating region of a synchronous generator are determined by factors relating to the operating conditions and the system capacity[1]. Operation in the reactive-generation region (ie operation with lagging power factor) is usually limited by the maximum power output of the turbine and the heating of the rotor and stator. In the reactive absorption region (ie operation with leading power factor) the restriction is set mainly by stability considerations. The stability limit of a synchronous generator is generally accepted to be defined by two considerations: its steady-state stability and its transient stability. The steady-state stability of a generator is its ability to maintain synchronism in the absence of a large disturbance to the system. This is in a dynamic mode under the influence of the governor and excitation control systems and instability is evident either by a monotonic increase in the rotor angle or the growth of oscillations of increasing amplitude. The former occurs as a result of insufficient synchronising torque and the latter is due to insufficient damping torque. Conventional automatic voltage regulators (AVRs) are tuned to minimise this instability[2].

The transient stability of a synchronous generator is its

ability to regain and maintain synchronism following a severe and sudden disturbance such as that caused by system faults or switching. Unlike the steady state limit, the transient stability limit is not well defined as it is dependent on the nature of the disturbance. Generally, the transient stability limit is defined as the maximum power that the generator can deliver without the loss of stability when subjected to the most onerous disturbance likely to occur in practice. This disturbance is usually accepted to be a symmetrical three-phase short circuit at the high voltage terminals of the generator transformer and is the test used in this study.

The transient stability problem is basically one of power balance, in that the exported power from the generator is severely limited during short-circuit conditions. The most obvious technique to redress this imbalance is either to reduce the mechanical input or increase the electrical output of the generator during and immediately following the fault occurrence. Various techniques have been employed to achieve this means, the mechanical input being reduced by fast governing[3,4] or fast valving[5,6] techniques. However, the effect of the entrained steam upon the system generally produces a slower response than that exhibited by the action of direct electrical control of the generator and is usually considered as secondary to this control. The electrical output of the generator may be increased by the use of braking resistors and reduction in fault clearing time. Further improvements in performance may also be made by the use of power factor correction[7], phase-shift insertion[8] and tie-line reactance control[9]. Most current work,

however, seems to be directed towards the use of excitation control as a means of increasing the transient stability limit. By this means it is possible to implement some of the ideas of modern control theory particularly with the introduction of fast excitation systems with increased ceiling voltage[10].

1.2 OBJECTIVES OF EXCITATION CONTROL

The primary objective of the excitation control system is to maintain the terminal voltage of the machine constant to within 0.5% of its nominal value under all acceptable steady loading conditions. This essential condition can be satisfied by the use of terminal voltage error feedback through a high gain amplifier tuned for steady state stability as described above. This form of feedback produces an inherent improvement in the steady state stability margin to a value well in excess of the uncontrolled value obtained with constant excitation[11,12] thus satisfying the second objective of the excitation controller. The third objective, which forms the subject of this study, is an improvement in the transient stability limit. This may be brought about by the addition of auxilary feedback signals as discussed in succeeding chapters.

1.3 DIRECT DIGITAL CONTROL

Recent years have witnessed rapid advances in digital technology resulting in significant reductions in digital component

costs and improvements in reliability[13]. This has promoted an increased interest in digital techniques in many areas of electrical science, one notable area being that of control systems. This development has been facilitated by the advent of small, fast minicomputers and microprocessors which are able to realise the sophisticated control functions of modern control philosophy.

Digital computers have been used in power stations for some time[14,15] to perform a variety of tasks. The majority of computer installations to date serve in a supervisory capacity, carrying out such duties as data logging, monitoring variables and checking for alarm conditions, contingency evaluations and economic dispatch considerations. It is only in nuclear stations where computers have been used for direct digital control and the trend is to control the starting reactors, turbines and auxilary equipment with these techniques. In stations under construction the boilers will also be controlled by computer.

As a stand-alone controller, a micro-computer can give several advantages. It is able to handle complex computations in a highly stable and repeatable manner and is very flexible, changes in mode of control being made by reprogramming with little or no hardware modification. Such controllers can bring within the realm of realisation, sophisticated control functions as envisioned by recent developments in the field of optimal and adaptive control. Within the power station environment, the concept of dedicated digital controllers produces even more advantages. The localised digital control system may communi-

cate directly with the main station or area supervisory computer and control loop parameters are readily changed from a central control area. Furthermore, such a system enhances the data acquisition and recording facilities of the main computer as well as providing the flexibility of control mentioned above.

It is sound engineering sense that the system is organised such that overall supervisory control is centralised but the control functions are maintained in localised loops. This increases the system efficiency and reliability by providing immediate control of units and reducing the effects of a controller breakdown while still allowing direct digital interaction between control units. Such a facility is provided by the use of dedicated microprocessors for each control function, linked to a central supervisor computer. The laboratory digital control scheme for the model generating system has been developed in this manner with a supervisory PDP-11/20 computer and an I8080 microprocessor for excitation control. Facility has been provided for the inclusion of additional microprocessors into the network for the ultimate control of excitation and governing of a multi-machine system.

1.4 PRELIMINARY INVESTIGATIONS

The control philosophy which will be developed in Chapter 2 is of the state-feedback type and, in theory, could be implemented in conventional analogue form with a constant feedback gain matrix. Theoretical considerations show that system sensiti-

vity to certain feedback states is high[16] and attempts at practical implementation showed that the noise content of the analogue signals produced steady state instability in the system. The development of a digital control facility was instigated partly as a solution to this problem by the use of low-noise digital transducers and partly to allow the future implementation of control philosophies not applicable by conventional analogue means.

CHAPTER 2

OPTIMAL AND SUBOPTIMAL CONTROL

The philosophy of system optimisation as developed in modern control theory can be briefly stated as follows. A performance criterion in the form of an integral of a function of the state variables and the control functions is defined and is to be minimised over a specified interval of time. The general principle is to minimise the state variable transients without excessive control effort. If no penalty or limitation is placed on control effort, then minimisation of the state transients might demand an impractically large (theoretically infinite) control input. Mathematically, the problem may be stated that, given a dynamic system represented by the vector differential equation:

$$\dot{x} = f(x,u,t) \qquad (2.1)$$

where x is a vector of the state variables and u is a vector of the control functions, select a form of control that will minimise

$$\int_0^T L(x,u,t)\, dt \qquad (2.2)$$

which is the defined performance index. It can be shown from the principle of optimality (dynamic programming) or Pontryagin's principle that if

$$\underset{u(t)}{\text{Min}} \int_{o}^{T} L(x,u,t) = V(x,t)$$

then the optimal control u satisfies the partial differential equation

$$\underset{u^*(t)}{\text{Min}} \left[L(x,u,t) + \sum_{i=1}^{n} f_i(x,u,t) \frac{\partial V}{\partial x_i} \right] + \frac{\partial V}{\partial t} = 0 \qquad (2.3)$$

No general solution of the partial differential equation (2.3) exists and a solution may only be obtained in certain cases. For high order non-linear systems, a closed form of control is not realisable and the solution is usually in the form of an open loop control or a feedback control with time-variant gains. The same is true if the technique of variational calculus is used resulting in a two-point boundary value problem which, in general, can only be solved by iterative methods requiring successive integrations of the state and adjoint equations. Regardless of the derivation, an open loop or variable gain state feedback type of control is only suitable to systems which have fixed parameters and operating conditions and are subject to a given disturbance. Practically, this is not the case when controlling turbogenerator sets in power systems. In the following section a brief consideration will be given to the work of other authors in the field as an introduction to the control philosophy used in this study.

2.1 PREVIOUS STUDIES

Conventional control theory predicts that an improvement in the

stability of certain types of system may be made by the use of a derivative form of feedback - so called 'velocity feedback'. Early workers studied the effects of this form of feedback on synchronous generator systems and reported that an improvement in the steady state stability could be achieved[18] at the expense, however, of an increase in the magnitude of the first swing of load angle following a large disturbance[3]. Quite often the addition of these auxilary feedback signals was made by intuitive means or by the use of classical design techniques, such as the inverse Nyquist technique[10], not strictly applicable to non-linear systems. Despite these limitations, the use of a speed feedback signal has proved effective in improving the stability limits of a large hydro-electric system in Canada[19].

More recently, several authors[20-22] have employed some of the techniques of modern control theory to a linearised model of the generating system. Modern control techniques can produce an optimal control of the linear model by solution of the Matrix-Riccati equation (see section 2.2). The problem of controlling a linear time-invariant system with a quadratic performance index is generally known as the Linear Regulator problem and the technique of dynamic programming or application of Pontryagin's Maximum Principle produces a control law which is a linear combination of all the states if the control period extends to infinity. However, this control law is strictly optimal for only the linear system or, in practice, for the non-linear system under small disturbance conditions about the optimised operating point. This does not necessarily produce the greatest improvement in the transient stability

limit of the practical system. Furthermore, the effectiveness of many of the control laws developed by this technique has only been tested on the same mathematical models as those used for the optimisation, which is clearly not indicative of the operation of a real system.

As an alternative, some authors have tested the effectiveness of control laws derived in this manner by applying the control to either a non-linear mathematical model of the system[23] or to an experimental micro-machine model[24]. This assumes that, although the controller only gives optimal closed-loop system behaviour in a region close to the selected operating condition, this form of control will provide improved closed-loop performance over a much wider operating area. Under fault conditions, this controller is not found to be entirely satisfactory[23]. It provides faster settling time of the system variables but the amplitude of the first swing always exceeds that obtained using conventional methods of control using a voltage regulator alone. Thus the transient stability margin is reduced by the presence of these auxilary signals.

As mentioned above, if the performance of a power system is evaluated using the same mathematical technique as that used to determine the feedback gains, then the validity of results relating to large system disturbances is in doubt, particularly in the leading power factor region. One novel technique[25] attempts to overcome this problem and improve the control by evaluating a set of optimal feedback gains to satisfy a simple quadratic cost function at a matrix of grid points distributed

over the complex power plane. These gains are stored in a small computer and applied to a laboratory microalternator as the operating point changes. Using this control the steady state stability limit is extended but no results have been given for behaviour under fault conditions.

In the optimisation of controlled system behaviour discussed so far, it has been assumed that the system controlled is linear and that all the states of the system are available for feedback control. In a real system this is not usually the case and several authors have attempted to overcome this problem by minimisation of a performance index, assuming that only some of the states are available for feedback[26,27] and using observer methods to recreate the states not available for measurement[28]. Although the results show improvement over conventional control techniques, comparative tests have only been performed on linearised models and are thus not applicable to the large disturbance conditions. Some practical work has also been performed using this technique[29] but no fault studies are recorded.

The optimisation of controlled system behaviour using a non-linear model is an inherently difficult problem to solve. Use of the technique of variational calculus results in a two-point boundary value problem which can only generally be solved by iterative techniques, such as that of quasi-linearisation[30]. An alternative method is by the use of differential dynamic programming[31]. The solution obtained by either of these methods is generally in the form of an open-loop control or a feedback

control with time-variant gains. Such a control cannot be used directly for a real power system with varying parameters, operating conditions and subject to different disturbances. It is possible, however, that such controls may be useful in modified form for future work under the influence of computer control systems which are able to store large quantities of control information for varying conditions.

2.2 OPTIMAL CONTROL OF LINEAR SYSTEMS

In this section a brief consideration will be given to the optimal control of a linear system prior to determination of the control law for a non-linear system in the next section. Basically, the problem to be considered is the Linear Regulator problem.

Consider a linear system described by the vector matrix equation

$$\dot{x} = Ax + Bu \qquad x(t_o) = x_o \qquad (2.4)$$

An optimal control, u, is to be applied over the closed interval $t_o \rightarrow t_f$ such as to minimise the quadratic performance index

$$I = \int_{t_o}^{t_f} \left[x^T Q_1 x + u^T Q_2 u \right] dt + x^T(t_f) Q_3 x(t_f) \qquad (2.5)$$

where Q_1 and Q_3 are n x n positive semi-definite symmetric matrices and Q_2 is an m x m positive definite symmetric matrix, n is the dimension of the state vector, x, and m is the

dimension of the control vector, u. The Hamiltonian function[17] is given by

$$H = x^T Q_1 x + u^T Q_2 u + \lambda^T (Ax + Bu) \qquad (2.6)$$

Pontryagin formulated the necessary conditions for optimal control in terms of the Hamiltonian as follows:

$$\frac{\partial H}{\partial \lambda} = \dot{x} = Ax + Bu \qquad \text{(State Eqn)} \qquad (2.7)$$

$$\frac{\partial H}{\partial x} = -\dot{\lambda} = 2Q_1 x + A^T \lambda \qquad \text{(Adjoint Eqn)} \qquad (2.8)$$

$$\lambda(t_f) = 2Q_3 x(t_f) \qquad \text{(Transversality Eqn)} \qquad (2.9)$$

$$\frac{\partial H}{\partial u} = B^T \lambda + 2Q_2 u = 0 \qquad \text{(Control Eqn)} \qquad (2.10)$$

The optimal control is thus given by

$$u^* = -Q_2^{-1} B^T \lambda / 2 \qquad (2.11)$$

Substituting equation (2.11) into the state equation (2.7) gives

$$\dot{x} = Ax - BQ_2^{-1} B^T \lambda / 2 \qquad x(t_o) = x_o \qquad (2.12)$$

If it is assumed that

$$\lambda(t) = 2K(t)\, x(t) \qquad (2.13)$$

then, on differentiation

$$\dot{\lambda} = 2\dot{K}x + 2K\dot{x} \qquad (2.14)$$

Elimination of λ, $\dot{\lambda}$ and $\dot{x}$ by substitution of equations (2.12) to (2.14) in equation (2.8) gives

$$2(\dot{K} + KA + A^{T}K - KBQ_2^{-1}B^{T}K + Q_1)\, x = 0 \qquad (2.15)$$

Equation (2.15) is only true for all x if the following condition is satisfied:

$$\dot{K} = -KA - A^{T}K - Q_1 + KBQ_2^{-1}B^{T}K \qquad (2.16)$$

By comparison of equations (2.9) and (2.13) it may be seen that the boundary condition is given by:

$$K(t_f) = Q_3$$

and as Q_3 is symmetric, so is K(t). Equation (2.15) is the Matrix-Ricatti Equation and from equation (2.11) the optimal control is given by:

$$u^* = -Q_2^{-1}B^{T}Kx \qquad (2.17)$$

where the symmetric matrix K may be obtained from the solution of the Matrix-Riccati Equation. If the control interval is extended to infinity then, for a time-invariant system,

$\dot{K} = 0$ and the Matrix-Riccati Equation becomes:

$$KA + A^T K + Q_1 - KBQ_2^{-1}B^T K = 0 \qquad (2.18)$$

where A and B are now constant matrices. Thus K is now a constant symmetric matrix and the feedback law given by equation (2.17) is a linear combination of all the states of the system.

2.3 PERFORMANCE OPTIMISATION OF NON-LINEAR SYSTEMS

The forms of control produced for non-linear systems using optimal control theory are generally of the open loop variety. Thus the derived control is only optimal for certain system parameters and operating conditions when subjected to the specified disturbance. As previously mentioned, this form of control is not desirable for the control of turbogenerator sets in power systems.

In the case of linear time-invariant systems, optimised for a quadratic performance index, it has been shown that the optimal control law is a linear combination of all the states. As the state vector completely specifies the system condition of a linear system, it would seem logical to assume that a sub-optimal control of a non-linear system could be achieved by linear feedback of some or all of the states, if the magnitudes of the non-linearities are not too great. If this is the case, then the performance of the system can be optimised by choosing

a set of optimum feedback gains. Thus the control problem now becomes a parameter optimisation problem.

The parameters of the feedback matrix are to be optimised to minimise an objective function which is chosen to be a quadratic performance index of the form

$$I = \int_0^T x^T Q x \, . \, dt \qquad (2.19)$$

where x is the state vector and Q is a positive semi-definite symmetric matrix. As the sub-optimal control is a linear combination of the state variables, the control u is given by

$$u = Kx \qquad (2.20)$$

The function minimisation subroutine used in this investigation is that used by Lee[32] which is based on the Quasi-Newton method of Gill and Murray[33]. The minimisation process consists of a systematic sequence of linear searches. The directions of search are determined using the first order sensitivity coefficients or gradients of the performance index. The sensitivity function, z, of the performance index with respect to an optimising parameter, α, is derived by partial differentiation of equation (2.19) with respect to the parameter:

$$z = \frac{\partial I}{\partial \alpha} = \int_0^T \frac{\partial}{\partial \alpha} (x^T Q x) \, dt \qquad (2.21)$$

The value of the performance index is evaluated at each point by solving the system equations and during each linear search, polynomial interpolation is used to locate the minimum in that direction. Linear searches are continued until either the convergence criterion is satisfied or reduction in the performance index is not achieved in a certain direction. A local search is then performed to ensure that the point arrived at is not a saddle point. The minimisation process is halted if the local search fails to reduce the performance index further and the feedback matrix values are printed. If the local search succeeds in reducing the performance index still further then the linear searches are continued.

As the single machine system studied is a non-linear system, it is expected that there are more than one local minimum of practical interest in the parameter space. Because of this, it is found to be convenient to have the feedback matrix and performance index values printed periodically during an optimisation run. As is expected, optimisation results show that the local minimum actually arrived at and the rate of convergence are influenced by the initial point chosen to start the function minimisation. However, as later results will show, the practical system response is sufficiently insensitive for these minor differences to be insignificant. The actual system equations used for the minimisation process and the performance index chosen are discussed in the following chapters.

CHAPTER 3

THE MODEL SYSTEM

All experimental results have been obtained from tests on a laboratory model of the electrical power system. For the test results to be applicable to an actual power system, the laboratory model should be representative of the type of system currently under development. The model system is based on a section of the British 400 kV network, modelling Pembroke Power Station in South Wales. This station is of the conventional thermal type with four turbogenerator units, each of 500 MW capacity, generating at 22 kV and connected through individual step-up transformers to the 400 kV station busbars. Two double-circuit transmission lines then connect the station to other points on the 400 kV network. This station is situated on a coastal site and transmission lines of the order of 110 KM in length connect the station to the grid. For these studies, the four identical machines are considered equivalent to a single machine whose per-unit parameters are identical to those of a single machine. This particular laboratory model system was developed by Martin[34] and the techniques have been well covered by many authors[35-38] so only a brief description will be given below. A diagram of the general system arrangement is given in Fig 3.1.

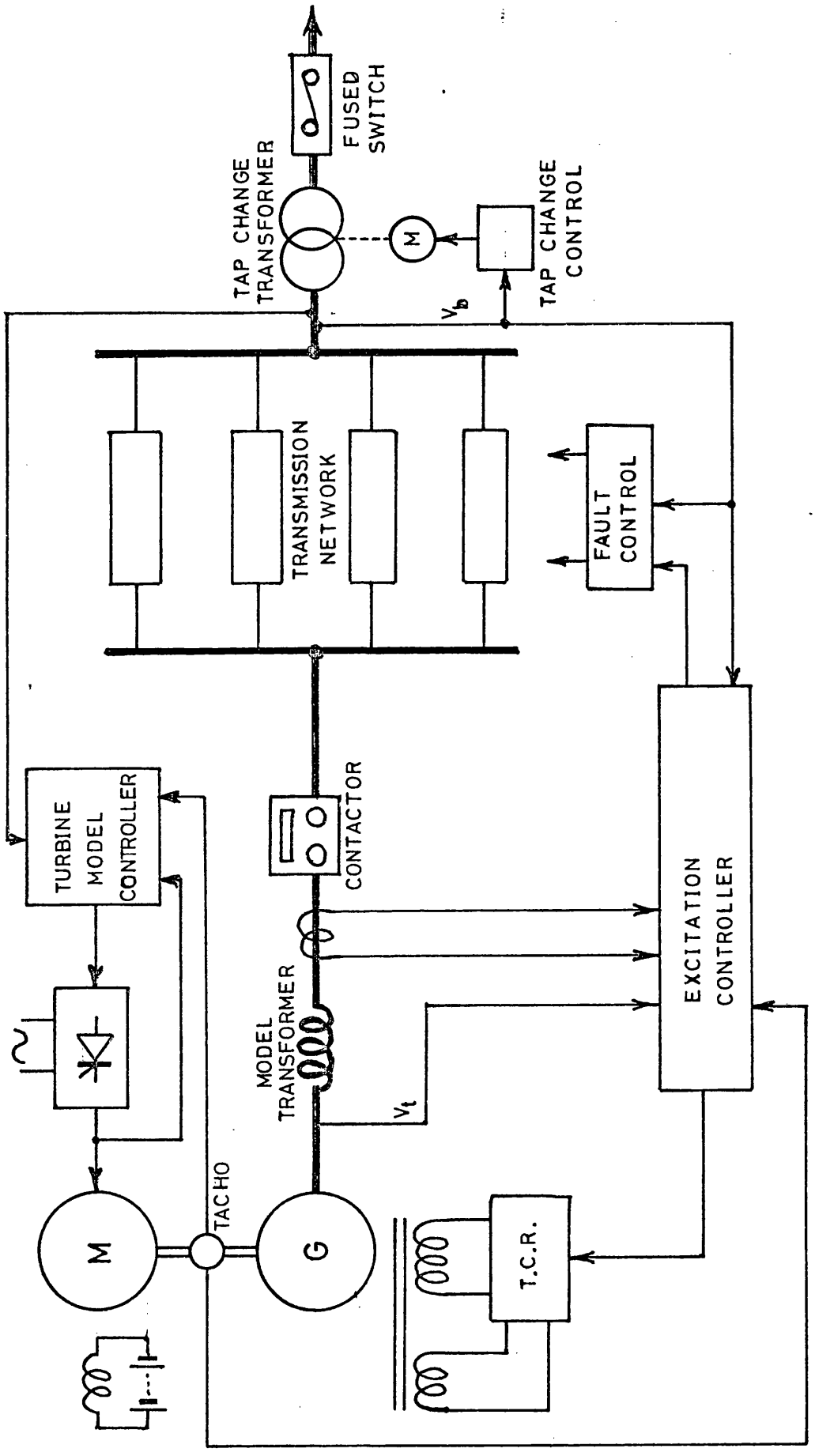


Fig. 3.1 Micromachine Model System General Schematic

3.1 THE MICROMACHINE MODEL

The heart of the model system is a synchronous generator constructed with the application of scaling factors to reproduce the characteristics of the larger machine being modelled. In order to achieve this equivalence, the per-unit electromagnetic and mechanical constants of both systems must be identical, requiring that the micromachine be a specially produced alternator incorporating several unusual features.

The micromachine used in this study is of the type described by Hammons and Parsons[36]. This machine is specially constructed using a high grade magnetic material and windings, dimensioned such that the per-unit leakage and mutual reactances and saturation characteristics are similar to those of the larger machine. Direct physical scaling does not produce a machine with identical per-unit characteristics as the winding resistances tend to increase disproportionately as the size is reduced. On the machine stator, this effect is overcome by using an increased slot depth to enable the copper cross-sectional area to be increased. This results in a machine which is overdimensioned in the generally accepted sense (ie the 3 kVA machine is built into a 30 kVA housing). On the rotor this technique is not possible as other parameters would be adversely affected. These high rotor resistances would lead to low transient and subtransient time constants and the dynamic performance of the machine would be changed. To overcome this problem, an external device must be used to regulate the effec-

tive value of the rotor resistances to a lower value. Such a device is generally known as a time constant regulator (TCR) and is described in Section 3.4.

The micromachine model is of the basic two-axis type with a three-phase armature winding and field windings on the direct and quadrature axes. In the current investigation, the q-axis field winding is not used and is only included in the machine design for generality. The circulating current paths in the rotor of the actual machine are simulated by damper windings on the direct and quadrature axes. Each rotor winding also has a shadow winding for use in connection with the TCR as described later. Ideally, these windings should have unity coupling to the primary windings and an interleaved winding pattern is employed to maximise this coupling ratio. The rotor is cylindrical, with four groups of equally spaced slots forming a four-pole configuration. Thus, the synchronous speed of the machine is 1500 rpm at a system frequency of 50 Hz. The machine is rated at 3 kVA, 0.8 pf and 230 V line voltage with the stator winding star connected.

3.2 THE TURBINE SIMULATION

During transient operation of fossil-fuel power systems, there is a large time constant associated with the entrained steam in the turbine and governor action has little effect on the first swing transient stability of a turbogenerator unit. However, since the problem during load change and short circuit

conditions is one of power imbalance, the action of the turbine and governor is important in the overall response of the system. During short-circuit conditions, the mechanical power input to the generator from the turbine is dissipated only in the heating losses in the resistances of the machine armature winding and the network between the machine and the fault. The excess power delivered by the turbine results in the acceleration of the rotor of the machine and thus particular attention must be paid to this aspect of the micro-machine model.

The large time constant associated with the entrained steam in the steam chest and reheater can result in time constants of several seconds and an apparent long delay between the closing of the steam valves and reduction of power input to the turbine. As described elsewhere, in view of this limitation, the work in this study has been focussed on the use of excitation control as a means of controlling transient stability limits. Thus, the common technique of ignoring governor action has been adopted for these first studies. In practice, this is performed by controlling the prime mover with a constant torque characteristic.

In the model system, the turbine is represented by a separately excited d.c. motor which is controlled by a single phase thyristor bridge rectifier unit from a 200 V supply. This was a commercial unit with controlled output voltage from zero to full voltage, from a convenient low level control signal. In

future work, this signal would be derived from a governor model but the present study required only that the 'turbine' torque be held at a constant value. In the loaded d.c. motor, the armature voltage and the rotational emf are of similar magnitudes with only a small voltage drop across the low armature resistance. Thus, a small change in rotor speed can produce a large change in armature current and a consequent large change in the torque produced. To overcome this effect, a constant current feedback controller around the d.c. motor is required to maintain constant torque. The constant current controller used in the model system was effective in maintaining fluctuations at the negligible value of 0.0034 p.u. at low frequencies and at unmeasurably small values at higher frequencies.

3.3 THE TRANSMISSION SYSTEM AND FAULT-THROWING EQUIPMENT

The generator is normally connected to the transmission system by a transformer which performs two functions. The first of these is to step up the generator voltage to the network level, and the second is to enable control of the transmitted reactive power whilst maintaining a constant generator terminal voltage. In the model system, the generated voltage is the same as that of the transmission network and so a unity turns ratio transformer would be required. A similar problem to that of the micromachine is encountered in the scaling process, in that the winding resistances tend to be too high in the model trans-

former. In view of the unity turns ratio required, a different approach is adopted. The reactances of the generator transformer are modelled by fixed inductors and small voltage adjustments are made by a tap-changing transformer at the infinite busbar end of the transmission line. The infinite busbar itself is modelled by a separate 330 kVA distribution transformer with an internal impedance of 0.0015 p.u. which is considerably less than the 0.014 p.u. impedance of the transmission network. The tap-changing transformer is similarly over-sized.

The transmission network itself has been simulated using four cascaded nominal π networks. The model is reported to give adequate modelling at normal system frequencies[34], although small errors are introduced at higher transient frequencies. For the purposes of the present investigation, the high frequency transients are not of interest.

The standard comparative fault test used in this study is a full symmetrical three-phase short circuit at the high voltage terminals of the generator transformer. In order to simulate actual system conditions under faulting and fault clearing conditions, semiconductor switches have been used to both apply and remove system faults with simultaneous fault application on all three phases but with fault clearance at the current zero in each phase. The general arrangement of the model fault throwing equipment is shown in Fig 3.2 and the final successful equipment involved the use of back-to-back thyristor pairs as the switches A and B in Fig 3.2. Facilities

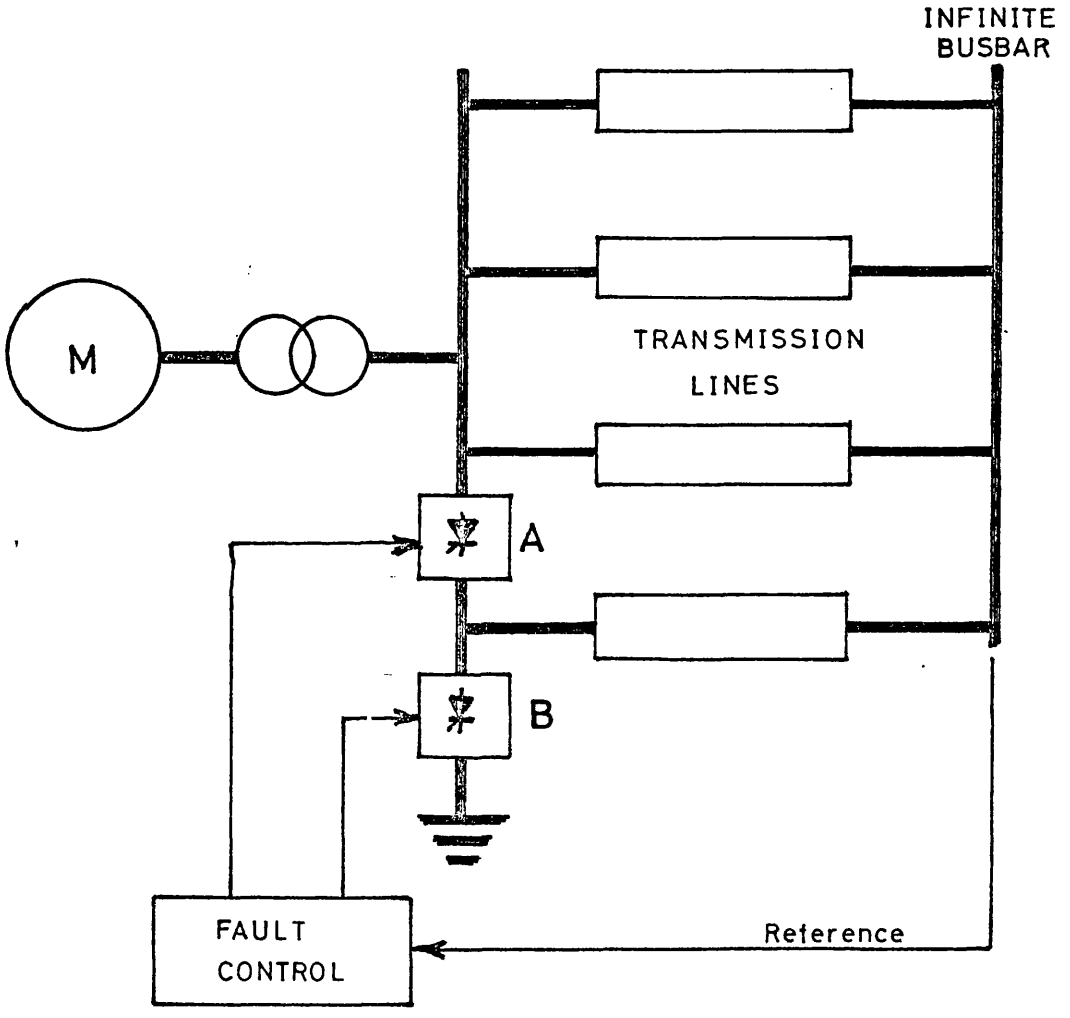


Fig. 3.2 Model System Fault-throwing Arrangement

were incorporated to allow a fault duration of from 40 mS to 240 mS, in increments of 20 mS, with the fault applied at a positive-going zero of one phase of a 50 Hz reference source. Then, with reference to Fig 3.2, device A performs the function of a circuit breaker controlling one transmission network at the station busbar, while device B throws a 3-phase short circuit on the same line. Suitable locations of devices of the type A and B allow the fault to be located at various points along the model transmission network and also permit various ratios of pre- to post-fault impedance. In the simple case of full system recovery, with the pre- and post-fault impedances equal, device A of Fig 3.2 is closed continuously, with all four lines in the circuit and the 3-phase fault is applied and cleared by device B. For the more realistic case of a fault followed by clearance of one or more paralleled transmission circuits, the fault is again applied with device B but it is cleared at the generator end of the transmission network by device A. Once device A has been opened, the fault can be completely cleared by device B at some arbitrary short time after the opening of device A. During the period of time between opening devices A and B, the fault is fed directly from the busbar and its presence can have no influence on the system.

3.4 THE EXCITER AND TIME-CONSTANT REGULATOR

It is well known that the field time constant of the micro-

machine is well below the value for a large turboalternator and that special arrangements in the form of a so-called time constant regulator must be incorporated to overcome this problem. The necessary power amplifier has, in the past, taken the forms of a rotating machine and a phase-controlled thyristor circuit. Early experience at Bath involved both phase controlled and chopper thyristor circuits and associated difficulties with waveform distortion led to the introduction of a linear, d.c. coupled power transistor amplifier. The present equipment used is, in fact, a commercial hi-fi power amplifier and has recently been incorporated in micromachine model systems at other British universities. Use of a linear d.c. amplifier achieves a good frequency response and prevents the induction of switching frequencies into the system. The commercial amplifier has a voltage and current limit of 60 V and 12 A.

The reduction in the effective field resistance is achieved by means of feedback from the shadow winding coupled to the field winding as previously described. The actual system in use at Bath was developed by Martin and has been well described by him[34] and other authors[39,40]. The reduction in apparent field resistance causes an increase in the field transient time constant which is adjustable up to 10 seconds. In the present investigation, it is set to 7.3 seconds to correspond to the open circuit transient time constant of the machine at Pembroke.

3.5 MODEL SYSTEM EQUATIONS

The mathematical model used for generator simulation is obtained by applying Park's transformation[41,42] to the basic three-phase equations. This transformation produces a resolution of the stator variables into components along the pole and inter-pole axes, referred to as the direct ('d') and quadrature ('q') axes, as well as a component from the zero-sequence variables. The latter component is not represented in the analysis below as it is not of significance in the operations considered. Certain simplifying assumptions are made as follows:

1. Hysteresis effects are neglected

2. The current in any winding is assumed to cause an m.m.f. wave which is sinusoidally distributed in space around the air gap

3. A component of m.m.f. acting along the direct axis is assumed to produce a sinusoidally distributed flux wave in the direct axis only, and similarly for the quadrature axis.

The machine equations in per-unit form can be shown to be as follows[43]: Flux equations:

$$\psi_d = -x_d i_d + x_{ad} i_{fd} + x_{ad} i_{kd} \quad (3.1)$$

$$\psi_{fd} = -x_{ad} i_d + x_{fd} i_{fd} + x_{fkd} i_{kd} \quad (3.2)$$

$$\psi_{kd} = -x_{ad} i_d + x_{fkd} i_{fd} + x_{kd} i_{kd} \quad (3.3)$$

$$\psi_q = -x_q i_q + x_{aq} i_{kq} \quad (3.4)$$

$$\psi_{kq} = -x_{aq} i_q + x_{kq} i_{kq} \quad (3.5)$$

Voltage equations:

$$v_d = p\psi_d/\omega_o - \omega_m \psi_q/\omega_o - r_a i_d \quad (3.6)$$

$$v_{fd} = p\psi_{fd}/\omega_o + r_{fd} i_{fd} \quad (3.7)$$

$$v_{kd} = p\psi_{kd}/\omega_o + r_{kd} i_{kd} \quad (3.8)$$

$$v_q = p\psi_q/\omega_o + \omega_m \psi_d/\omega_o - r_a i_q \quad (3.9)$$

$$v_{kq} = p\psi_{kq}/\omega_o + r_{kq} i_{kq} \quad (3.10)$$

The winding self reactances may be expressed in terms of the mutual and leakage reactances as follows:

$$x_d = x_{ad} + x_{ld} \quad (3.11)$$

$$x_{fd} = x_{ad} + x_{lfd} \quad (3.12)$$

$$x_{kd} = x_{ad} + x_{lkd} \quad (3.13)$$

$$x_q = x_{aq} + x_{lq} \quad (3.14)$$

$$x_{kq} = x_{aq} + x_{lkq} \quad (3.15)$$

The values of these reactances are obtained from the design

data of the Pembroke machines. The air-gap torque equation is

$$T_e = \psi_d i_q - \psi_q i_d \tag{3.16}$$

and the equation of motion of the rotor is given by

$$Mp^2\delta = T_m - T_e - K_d p\delta/\omega_o \tag{3.17}$$

and the machine terminal voltage by

$$V_t = \sqrt{v_q^2 + v_d^2} \tag{3.18}$$

Magnetic saturation effects are neglected as it has been shown[43] that saturation has little effect on the transient stability of the machine.

Making the usual assumption that x_{fkd} is equal to x_{ad}, the machine parameters may be more conveniently expressed in terms of the transient and subtransient parameters:

$$x'_d = x_{ld} + \frac{x_{ad}x_{lfd}}{x_{ad} + x_{lfd}} \tag{3.19}$$

$$x''_d = x_{ld} + \frac{x_{ad}x_{lfd}x_{lkd}}{x_{ad}x_{lfd} + x_{ad}x_{lkd} + x_{lfd}x_{lkd}} \tag{3.20}$$

$$x''_q = x_{lq} + \frac{x_{aq}x_{lkq}}{x_{ad} + x_{lkq}} \tag{3.21}$$

$$T'_{do} = \frac{x_{ad} + x_{lfd}}{\omega_o r_{fd}} \tag{3.22}$$

$$T'_{d} = \frac{1}{\omega_o r_{fd}} \left(x_{lfd} + \frac{x_{ad} x_{ld}}{x_{ad} + x_{ld}}\right) \tag{3.23}$$

$$T''_{do} = \frac{1}{\omega_o r_{kd}} \left(x_{lkd} + \frac{x_{ad} x_{lfd}}{x_{ad} + x_{lfd}}\right) \tag{3.24}$$

$$T''_{d} = \frac{1}{\omega_o r_{kd}} \left(x_{lkd} + \frac{x_{ad} x_{ld} x_{lkd}}{x_{ad} x_{ld} + x_{ad} x_{lkd} + x_{ld} x_{lkd}}\right) \tag{3.25}$$

$$T''_{qo} = \frac{x_{aq} + x_{lkq}}{\omega_o r_{kq}} \tag{3.26}$$

$$T''_{q} = \frac{1}{\omega_o r_{kq}} \left(x_{lkq} + \frac{x_{aq} x_{lq}}{x_{aq} + x_{lq}}\right) \tag{3.27}$$

The model formed from the preceding equations is of seventh order and takes account of the decay of stator flux linkages which give rise to power frequency oscillations in the machine voltages and currents. Because of these power frequency components, a small time step would be required for the numerical integration of the machine equations. In order to increase computing efficiency, particularly for the multi-pass optimisation studies, the decay in the flux linkages and the speed variations are ignored. After some algebraic manipulation[43] the fifth order machine model is obtained:

$$pe''_{d} = \left[(x_q - x''_q) i_q - e''_d\right] / T''_{qo} \tag{3.28}$$

$$pe'_{q} = \left[V_f - (x_d - x'_d) i_d - e'_q\right] / T'_{do} \tag{3.29}$$

$$pe''_{q} = \left[e'_q - (x'_d - x''_d) i_d - e''_q\right] / T''_{do} \tag{3.30}$$

$$e''_d = v_d + r_a i_d - x''_q i_q \tag{3.31}$$

$$e''_q = v_q + r_a i_q + x''_d i_d \tag{3.32}$$

$$T_e = e''_d i_d + e''_q i_q - (x''_d - x''_q) i_d i_q \tag{3.33}$$

The transmission line equations are obtained by ignoring transformer magnetising and line charging currents and some of the less important transient terms[43]:

$$v_d = V_b \sin \delta + R_t i_d - X_t i_q \tag{3.34}$$

$$v_q = V_b \cos \delta + R_t i_q + X_t i_d \tag{3.35}$$

The parameter values of the model system and of Pembroke Power Station are given in Table 3.1.

	Parameter	Model System	Pembroke Station
	Base Rating	3.45 KVA	2352 MVA
Generator	r_a	.00790 p.u.	.00310 p.u.
	x_d	2.80 p.u.	2.80 p.u.
	x_q	2.72 p.u.	2.72 p.u.
	x'_d	.361 p.u.	.362 p.u.
	x''_d	.180 p.u.	.230 p.u.
	x''_q	.220 p.u.	.220 pu
	T'_{do}	7.30 sec	7.30 sec
	T'_q	.94 sec	.945 sec
	T''_{do}	.0280 sec	.0314 sec
	T''_d	.0140 sec	.020 sec
	T''_{qo}	.116 sec	.116 sec
	T''_q	.00936 sec	.00936 sec
	H	4.44 p.u.	4.44 p.u.
	SCR	.416 p.u.	.440 p.u.
Transformer	R_T	.0130 p.u.	.00490 p.u.
	X_T	.157 p.u.	.157 p.u.
Line	R_L	.00794 p.u.	.00794 p.u.
	X_L	.103 p.u.	.109 p.u.
	B_L	.124 p.u.	.125 p.u.

TABLE 3.1 MODEL SYSTEM PARAMETERS

CHAPTER 4

OPTIMISATION RESULTS

The determination of a sub-optimal state feedback law was performed off-line on the main University ICL System 4 computer using subroutines developed by Lee[32] and Chana[44]. It is anticipated that future work may use the facilities of the PDP-11/20 computer for either on-line or off-line optimisation particularly as it has recently been equipped with a real-time monitor system[45] offering foreground/background programming capabilities.

4.1 FEEDBACK STATE SELECTION

The selection of states for inclusion in the feedback control depends on two main factors: the availability of the states for measurement and the sensitivity of the feedback law to each of the various states. To some extent the state variables of the system depend on the formulation of the system equations and these may be rearranged to produce a state vector whose components may be conveniently measured in the practical case. Consideration must be given during the rearrangement process to the sensitivity of the control to the elements of the state vector, as states of lesser significance in the feedback may be ignored in the formulation of a sub-optimal controller. Sensitivity studies on the system

equations in the preceding chapter show[32] that the following states are of greatest significance in the state feedback control:

δ	the load (or rotor) angle of the synchronous machine
$p\delta$	the transient velocity (first time derivative of δ)
e'_q	the voltage behind transient reactance (q-axis)

As these states may be conveniently measured in the practical system, they were incorporated in the state feedback optimisation process.

4.2 THE PERFORMANCE INDEX

In the optimisation process the performance index must be chosen to adequately represent the required performance of the system being optimised. For the power system under consideration, there are several requirements of system operation which are reflected in the choice of performance index. The most important requirement is that satisfied by the existing AVR control system, that the machine terminal voltage should recover as quickly as possible following a severe fault. Secondly, the control should produce a high first swing stability margin by keeping the load angle swing as small as possible. Thirdly, the subsequent oscillations of load angle should be rapidly damped out in a controlled manner. Such a control implies a minimisation of the terminal voltage and load angle transients and a per-

formance index suiting these requirements is of the form:

$$I = \int_0^T A_1 (\Delta V_t)^2 + \frac{A_2}{0.1 + t} (\Delta\delta)^2 \, dt \qquad (4.1)$$

where A_1 and A_2 are weighting factors. The form of this performance index is such that greater importance is placed on the first swing of load angle and the relative weightings of the load angle and terminal voltage responses may be changed. Without an exact specification the choice of performance index is rather subjective but a good compromise is found with the values $A_1 = 1$ and $A_2 = 5$.

4.3 LOW GAIN FEEDBACK

As stated in Section 1.4, the addition of state feedback signals derived from analogue transducers to the high gain input of the exciter caused instability of the system. Thus, preliminary investigations, both analogue and digital, were performed with the additional state signals added in after the high gain portion of the exciter as shown in Fig 4.1. The exciter is represented by a second order model and selection of the time constants enables representation of thyristor or a.c. excitation systems. The forward transfer function represents the main regulator and exciter and the feedback transfer function represents the stabilising derivative feedback. This is necessary because of the high forward gain requirement to achieve satisfactory steady state voltage regulation. The saturation of the exciter is represented by the limiters shown. It should

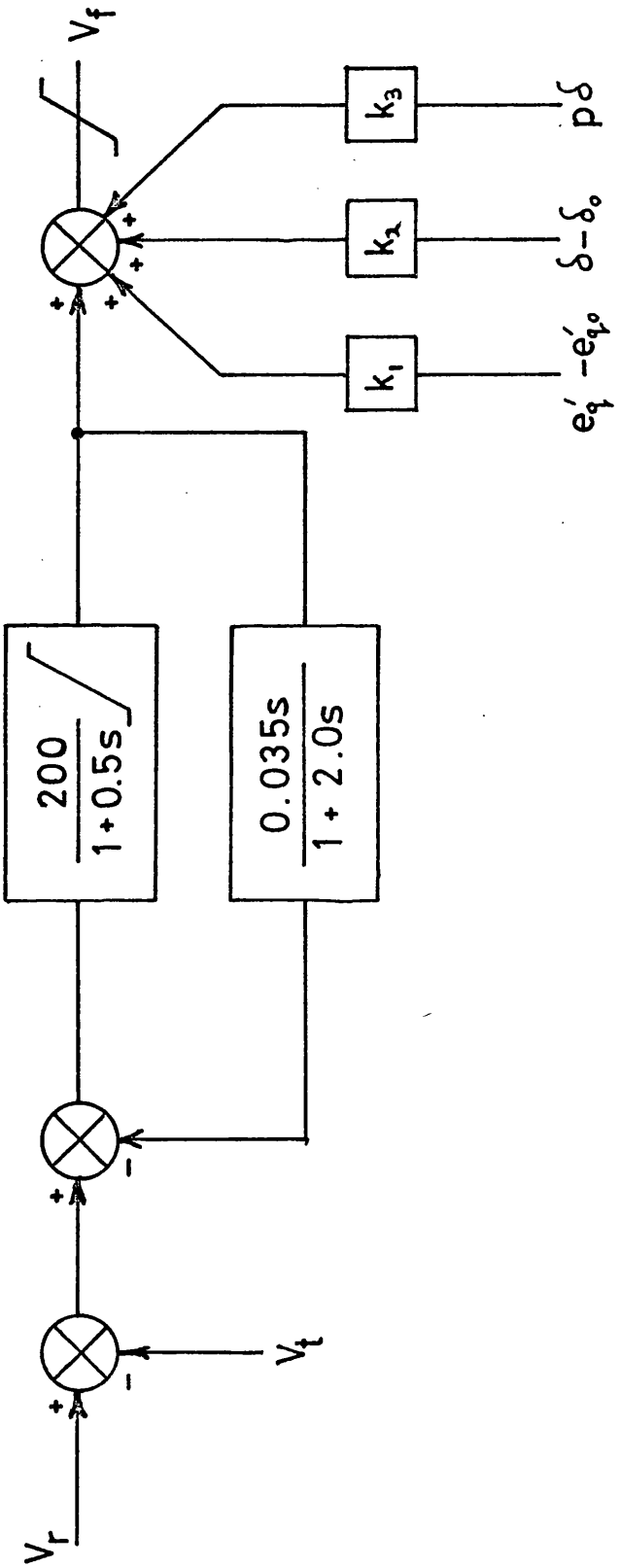


Fig. 4.1 Model Excitation System (Low gain state feedback)

be noted that this configuration could not be used on a full-size power system as the addition of the state feedback signals would be at the megawatt level. The exciter parameters are given in Table 4.1.

K_a	forward gain	200
K_f	stabilising feedback gain	0.035
T_e	exciter time constant	0.5 sec
T_f	stabilising feedback time constant	2.0 sec
$V_f(max)$	maximum main field voltage	6.87 pu
$V_f(min)$	minimum main field voltage	-6.87 pu

Table 4.1 Model Excitation System Parameters

Using this configuration and the optimising methods previously described, the system performance was optimised to minimise the performance index given above. Optimisation was performed for system performance following a symmetrical three-phase short circuit at the high voltage side of the generator terminals of duration 140 mS. The system was assumed to be operating at its normal condition at a power of 0.85 pu, with the terminal voltage set to 1.0 pu and unity power factor at the infinite busbar. The pre-fault and post-fault line impedances were set to be equal. Under these conditions the sub-optimal control law was found to be

$$V_i = -20.0\ \Delta e'_q - 5.9\ \Delta\delta + 2.5\ p\delta \qquad (4.2)$$

4.4 HIGH GAIN FEEDBACK

In a real power system the additional state feedback signals would have to be incorporated at the relatively low power input to the exciter as shown in Fig 4.2. This now represents an entirely realisable practical control system. Using the optimisation techniques previously described and for the standard fault condition and normal operating point described in Section 4.3, the sub-optimal feedback law is found to be:

$$V_i = -0.564\ \Delta e'_q - 0.4613\ \Delta\delta + 0.0027\ p\delta \qquad (4.3)$$

The states are fed back as state differences (eg $\Delta\delta$) to avoid the state feedback signal offsetting the steady state operating point.

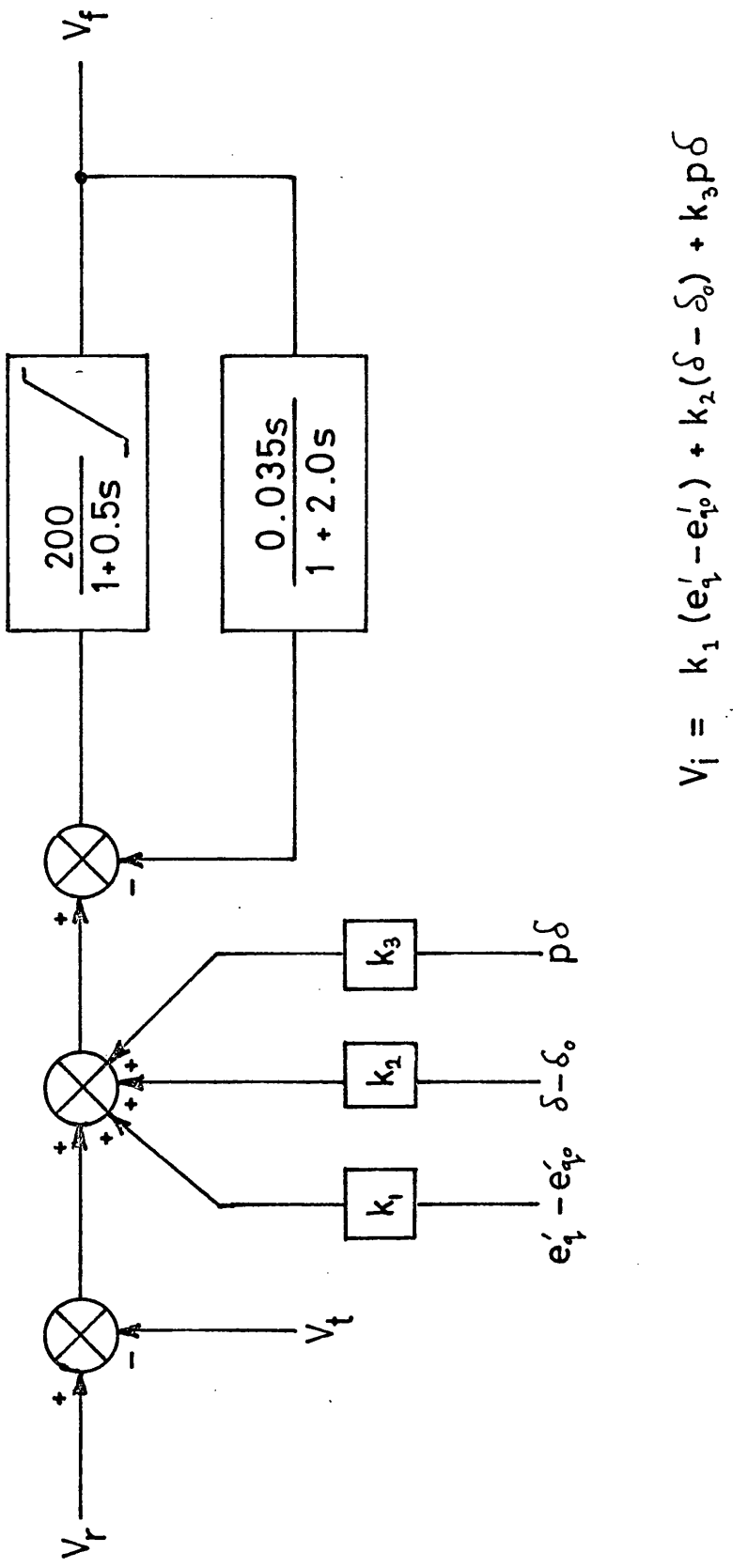


Fig. 4.2 Model Excitation System (High gain state feedback)

CHAPTER 5

THE COMPUTER SYSTEM

The digital control system described in this chapter comprises a supervisory computer and one microprocessor for excitation control of a single machine generating system. However, both the hardware (equipment) and software (programs) have been configured to allow the inclusion of additional microprocessors where necessary for the eventual control of excitation and governing of the two-machine generating system which exists in the power laboratory.

5.1 GENERAL SYSTEM DESCRIPTION

Preliminary investigations into digital excitation control using a mini-computer interfaced directly to the system model indicated that a large amount of processor time was spent in transducer servicing and data transfer operations. It was decided that these functions could be readily performed by one or more dedicated microprocessors which would service transducers, perform simple arithmetic operations on the data and transmit this data to the minicomputer which would then have more time for optimisation of the overall control strategy involved in the control of both excitation and governing of all four turbo-alternators in a power station.

The digital excitation controller for the micromachine model has two processing units, a PDP-11/20 computer which generates the main supervisory and control functions and an Intel I8080 microprocessor system which acts as a data collecting and filtering unit and which also performs local peripheral supervision and servicing functions. The general layout of the system is shown in Fig 5.1. The PDP-11 is remote from the micro-alternator and is linked by a data transmission system and interface module to the microprocessor which is physically close to the machine under control. The rotor angle and rotor transient velocity signals are generated in direct digital form by specialised transducers eliminating the noise problems inherent in analogue units. Other states are fed into the microprocessor via suitable transducers and analogue to digital converters. Within the microprocessor, a simple digital filtering algorithm is employed to remove the steady state offsets and to prescale the variables for input to the PDP-11/20. The variables are then transmitted through the digital data link to the PDP-11/20 which computes the feedback control function. This control information is transmitted back to the microprocessor and applied as the excitation control through a digital to analogue converter and the time constant regulator. The PDP-11/20 minicomputer has a versatile interrupt priority structure, which allows the implementation of multi-level processing priorities. Under steady state and small disturbance conditions no control effort is required of the PDP-11 which is free to perform background data-logging tasks. When a disturbance occurs which requires high utilisation of both processors, these

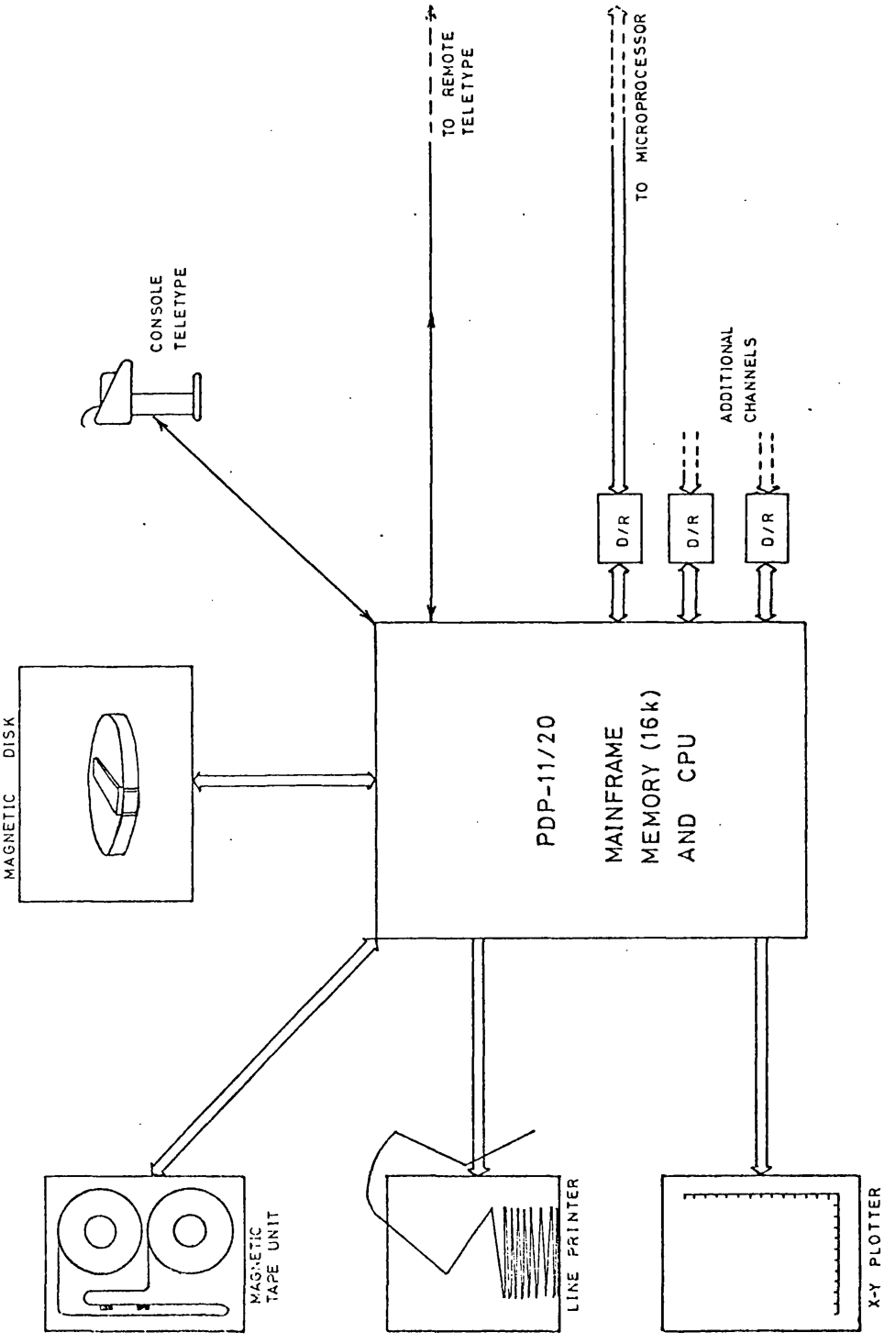


Fig. 5.1 (Part 1) Digital Control System General Schematic

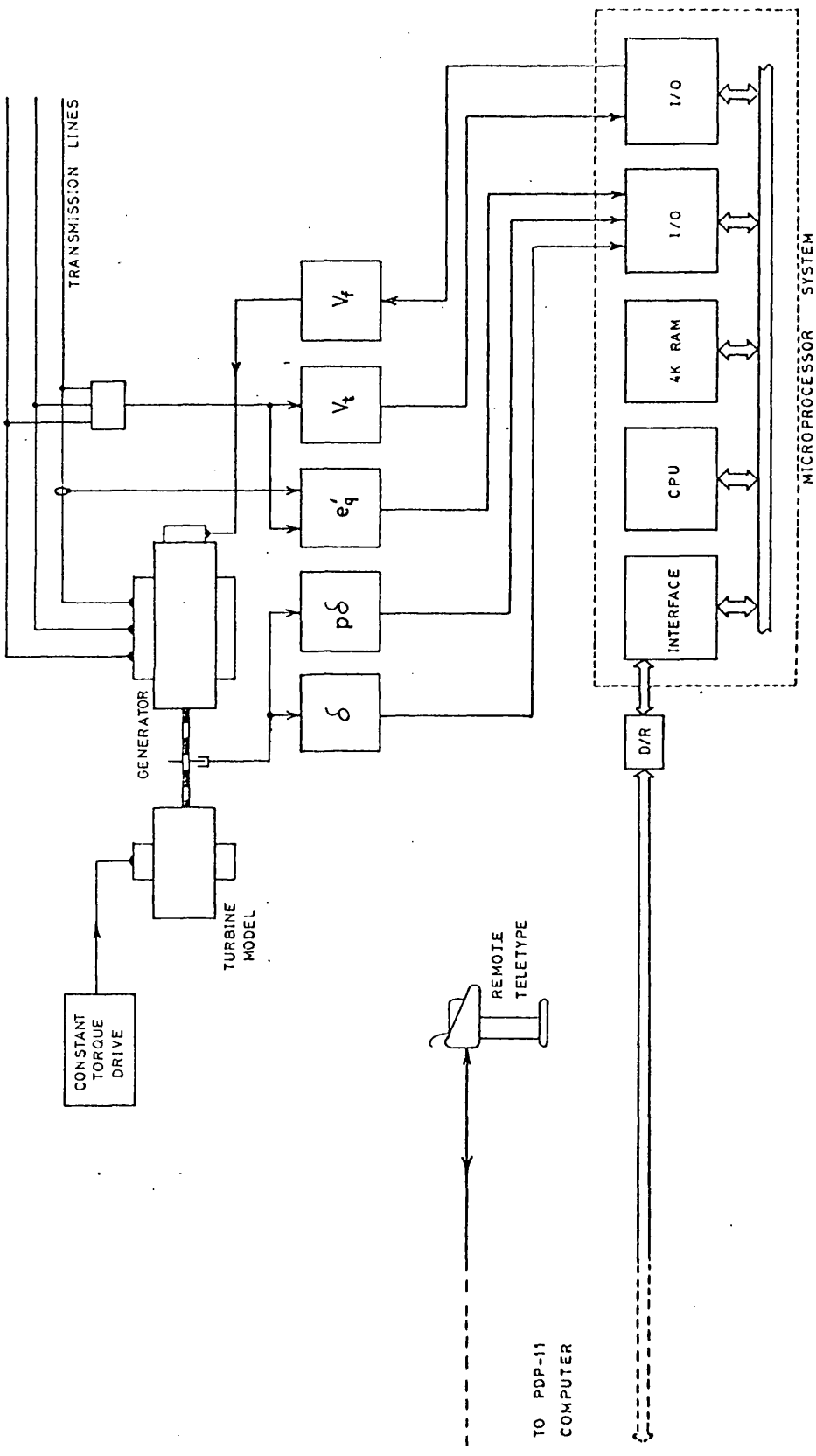


Fig. 5.1 (Part 2) Digital Control System General Schematic

background tasks are suppressed and high priority is given to control processing. System security is maintained by the microprocessor checking a 'time-out' situation from the PDP-11/20 and defaulting to a simpler control algorithm in the event of loss control information from the PDP-11 (caused by noise or breakdown of the link).

5.2 THE PDP-11/20 COMPUTER

The heart of the PDP-11/20 computer is the central processing unit (CPU) which has 20k* of 16-bit core memory and a KE-11A hardware arithmetic unit. The latter unit provides for fast multiplication and division of 16-bit numbers as well as shifting and normalisation functions. A typical operation is performed in 4.5 μS. The CPU itself has a basic cycle time of 2.3 μS for register to register operations and 5 to 6 μS for memory to memory operations. The system is ideally suited to real-time control as the computer is built on a Unibus** structure[46] and maintains a versatile priority interrupt system. This interrupt facility functions via vectored interrupts, which give an inherently faster response than a polled system.

The main storage medium of the computer is an RK05 magnetic disk pack which has a 1.2 Megabyte storage capacity and a maximum access time of 40 ms. Data transfers to and from the disk are performed in the direct memory access (DMA) mode and, once a

---

* 1k = 1024 words

** DEC trademark

track is accessed, may proceed at a rate limited only by the speed of the intermediate hardware, typically one word per 500 nS when transferring to memory. Furthermore, because of the Unibus structure, these inter-device transfers may take place without the supervision of the CPU which may continue its normal processing sequence. The disk unit is used for the storage of all system and user programs and for the immediate storage of test results during real-time control tests. A storage backup facility is provided by the TU10 industry-compatible magnetic tape unit. This enables test results and backup copies of programs to be transferred to ½ inch magnetic tape for safe and permanent storage. Test results may also be output to the type 26000/A4 X-Y plotter for visual interpretation or, alternatively, all results and programs may be listed on a medium speed (approximately 120 lines/minute) Centronics line printer. The main system command unit is an ASR-33 teletype which may be either local to the PDP-11 system or in a remote location as shown in Fig 5.1. Interfacing with the transmission link to the microprocessor and generating system is performed via DR-11A (or DR-11C) general device interface units as described in the Appendix, Section 3.

5.3 THE I8080 MICROPROCESSOR

The microprocessor system is based on an Intel 8080 CPU which was first introduced in 1971. The device is an 8-bit NMOS microprocessor which has a 2 μS instruction cycle and 74 basic

instructions. It has the facility to address up to 64k bytes of memory without the need of an external address register and can also directly address up to 256 input and output ports. This particular device was adopted into the digital control system in late 1973 and at the current time (1977) could be replaced with advantage by one of the more recently introduced microprocessors. The major disadvantage of the I8080 microprocessor is that it has an eight bit word limiting the accuracy of single-word data to $\pm$ 0.4% in two's complement arithmetic. The instruction set is such that operations may be conveniently performed on two bytes to produce 16-bit accuracy ($\pm$ 0.003%) at the expense of a typical addition being five times slower than the single byte case. This drawback is overcome in the recently introduced 12 and 16 bit microprocessors. A further limitation is the absence of a fast hardware multiply/divide facility. Again this can be overcome by use of a microprocessor containing this facility or by the addition of a multiply/divider unit to the system. In the current system all multiplications by constants in the microprocessor are performed by pre-programmed shifting and addition algorithms while more complex variable by variable multiplications are performed in the PDP-11.

The microprocessor memory is a 4k random access memory (RAM) which has an access time of 700 nS. This access time is greater than the cycle time of the CPU (500 nS), causing the latter to wait for memory response in every operation which reads from or writes to memory. Typically, the program

execution speed could be increased by 10 to 30% if a faster memory unit is used. An input/output facility is provided by two cards each containing eight 8-bit input ports and four 8-bit output ports. These cards have been specially constructed to improve real-time system performance by the implementation of hand-shaking techniques as described in the Appendix, Section 6.

5.4 THE MICROPROCESSOR TO COMPUTER INTERFACE

The interface card between the I8080 Microprocessor and the PDP-11/20 computer is an important part of the laboratory digital control system, as it is this card which gives efficiency, flexibility and ease of operation to both on-line and off-line computer operations with respect to the microprocessor system. A functional block diagram of the interface, is given in Fig 5.2 and the operation of this device will be considered in terms of its four major modes of operation. A complete technical and operational description of the Interface Card is given in the Appendix, Section 2.

5.4.1 Programming and Direct Memory Access

At the time of purchase of the I8080 microprocessor (1973), there existed only three means of loading programs into the device memory. The first, and technically simplest, method was to load the program 'noughts' and 'ones' via switches connected to the data and address busses or in a slightly more sophisticated

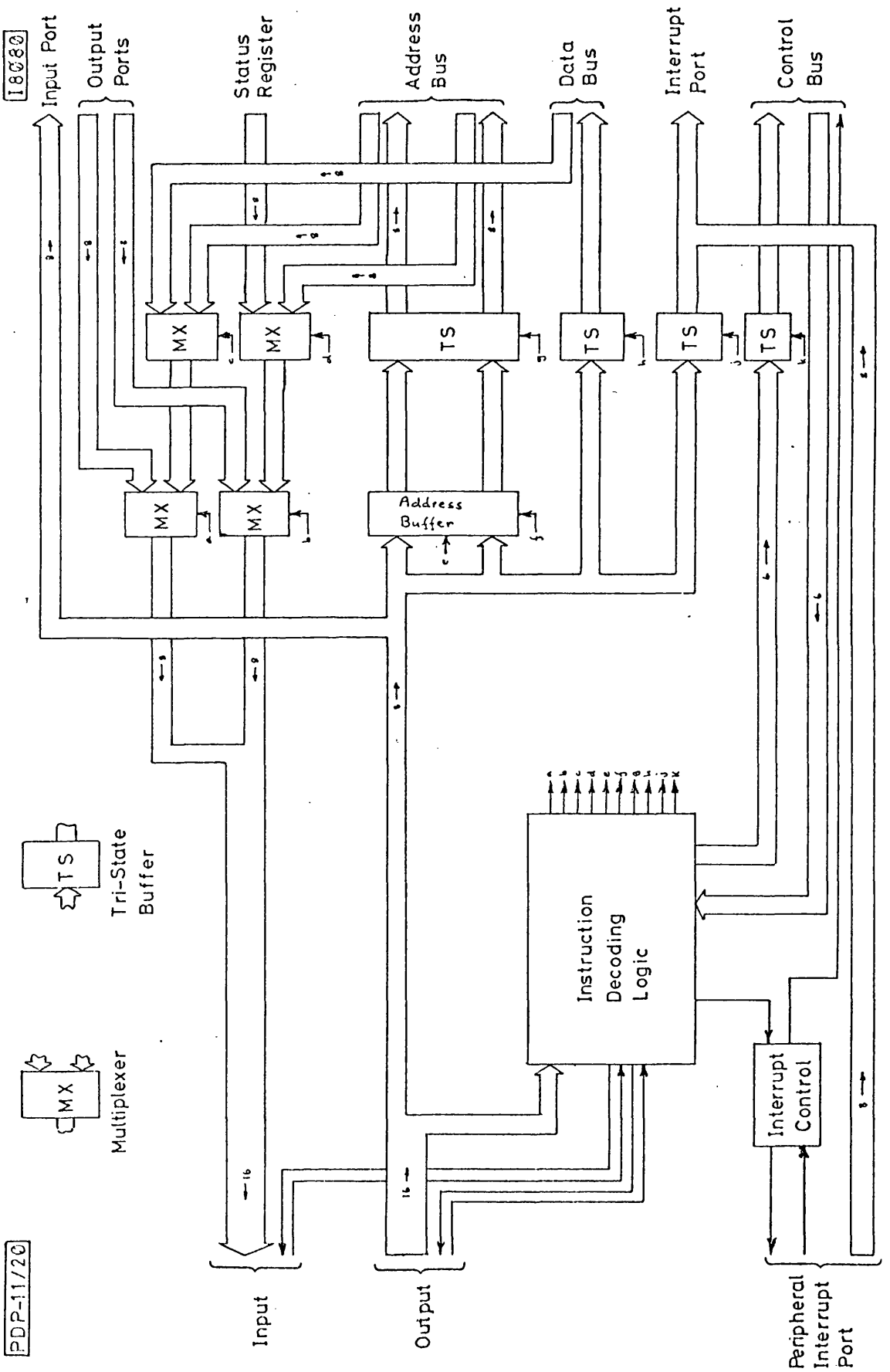


Fig. 5.2 Functional Block Diagram of PDP-11/20 to I8080 Interface Card

form to load from a paper tape reader arranged to drive these busses. The disadvantage of the switch loading method was obviously that it was very tedious for anything but the simplest of programs. Loading from a paper tape suffered from the drawback that a new paper tape had to be prepared every time a small program change was made.

The second means of loading a program into memory was to load it by means of assembly language mnemonics converted by a resident assembler program from a teletype, either typed directly or obtained from a paper tape. This suffered from the disadvantage that the editor and assembler programs were slow to load into the microprocessor memory (typically 20 minutes each at 10 characters per second) and that they consumed relatively large amounts of the microprocessor memory (typically 3.5k for the assembler). Furthermore, all the processes discussed so far had to be repeated if a major programming fault destroyed the loaded program or it was necessary to disconnect the power to the system to make hardware modifications. The third method, which overcame these drawbacks, was to use programmable read-only memory arrays (PROMs). These however suffered from the disadvantage of being difficult to program and not flexible enough for program development. Thus, the first consideration of the microprocessor/computer interface was to provide some means of rapid programming overcoming all the abovementioned disadvantages.

This facility is provided by the direct memory access (DMA)

facility built into the Interface Card. Microprocessor programs may be stored or generated in the PDP-11/20 using all the high level facilities of the main computer (refer to Chapter 6 and the Appendix for details) and transferred to the microprocessor memory at high rates whenever needed. With reference to Fig 5.2, it can be seen that this is achieved by the Interface Card taking over control of the microprocessor data, address and some control busses and loading data from the PDP-11/20 directly into the microprocessor memory. Such operations can in fact be performed on a 'cycle-stealing' basis to change specific locations in the microprocessor memory while the microprocessor CPU is engaged on an internal operation. Single word transfer speeds can be very high but block transfer rates slow down to typically 100k bytes/second. However, there is no problem due to loss of microprocessor program with this facility as it can be immediately reloaded from the supervisory computer magnetic backup media. A further use of the DMA facility, to be discussed later, is that it aids program debugging by allowing the microprocessor memory to be examined by the PDP-11 at any point in the program operation.

5.4.2 Control and De-bugging

Once the PDP-11/20 computer had been provided with a means of loading programs into the microprocessor memory, it was a logical extension to provide a means of testing and debugging these programs by means of high-level interactive techniques provided via the console of the supervisory computer. In terms

of the hardware requirement of the Interface Card, this necessitated the PDP-11 computer having access to all required microprocessor control busses and status registers as well as access to the memory and input/output ports by the DMA facility discussed above. The Interface Card decodes the 16-bit instruction from the PDP-11 and performs the necessary operations on the microprocessor controls or returns the requested information as detailed in the Appendix. Generally, the facilities provided are: reading and changing memory locations; receiving data from or sending data to input/output ports; single stepping, running or halting a program; simulating interrupts and data transfers; and resetting all or part of the microprocessor system.

5.4.3 Interrupts

In its basic form, the I8080 microprocessor provides only one interrupt request facility. In order to make real-time control system as fast and as flexible as possible it is necessary that each transducer in the system is capable of generating an interrupt and that some form of priority structure is provided for processing the various peripheral interrupts. The Interface Card provides the base level of this peripheral interrupt priority structure and allocates highest priority to interrupts from the PDP-11. Lower priority interrupts are handled by the Peripheral Priority Card (Appendix, Section 7) which 'stacks' onto the Interface Card. The Interface Card also provides the PDP-11 with a means of simulating all other peripheral

interrupts - a useful aid to system debugging.

5.4.4 Data transfers

The Interface Card also handles the regular program-controlled data transfers between the PDP-11 and the I8080, data is transferred in 8-bit parallel form from the PDP-11 to the I8080 and in optional 8 or 16-bit parallel form in the reverse direction. Data transfers are acknowledged by a hardware 'handshaking' technique, i.e. a flag is set automatically when data is read by the receiving processor and this is arranged to provide an optional 'interrupt-when-done' facility in the sending processor providing for faster system operation.

CHAPTER 6

TRANSDUCERS AND MICROPROCESSOR PERIPHERALS

All transducers connected to the model generating system interface with the microprocessor which acts as a 'front-end' servicing, processing and control subsystem. Whenever possible transducers have been designed to be of the direct digital type to eliminate accuracy problems of analogue transducers and to facilitate more efficient interfacing with the digital control system. Complete technical and operational details of all transducer units are given in the Appendix and this chapter discusses only the philosophy and principles of operation.

6.1 ROTOR TRANSIENT VELOCITY

The transient velocity ($p\delta$) of the rotor of the synchronous machine system is an inherently difficult variable to measure. Electronic differentiation of the load angle signal (δ) tends to produce an extremely noisy signal, as is true of any differentiation process. Direct measurement of the transient velocity signal is made difficult by the fact that this signal, of maximum amplitude 2.0 rad sec^{-1}, is superimposed on the shaft synchronous speed of 314 rad sec^{-1} (when referred to a two-pole machine system). Thus a noise content of say 0.5% in the synchronous speed signal produces a noise content of 75% in the transient velocity signal. Such a large noise content

makes signal recovery by filtering almost impossible as well as introducing extra time constants into the control system. It was primarily this reason that led to the abandonment of measurement of rotor transient velocity by analogue means.

The essential features of the transient velocity transducer are that it should have an inherently high accuracy to overcome the noise problem and that any attachment to the machine shaft should be simple and robust. The requirements are met by the transducer developed. The attachment to the machine shaft consists of a circular disc with a single radial slot which is mounted on the shaft in an arbitrary position of rotation. It is expected that such an arrangement is robust enough to withstand practical environmental conditions as evidenced by field trials on more complex transducer arrangements[19]. The periphery of the disc is arranged to pass between the sensors of two photo-pickup units spaced diametrically opposite each other on fixed mountings. On a two-pole machine only one such pickup would be required. In the two pickup system a micrometer adjustment is provided for exact positioning of the pickup units such that they are diametrically opposed. This procedure is conveniently performed under the control of the computer by running a calibration program (see the Appendix, Section 16).

The principle of operation of the transducer is as follows. The transducer interacts with the microprocessor internal clock to time the period between the disc slot passing from one pickup to the next. A count is produced in a buffer in the

transducer unit which is proportional to this period and is fed to the microprocessor for subsequent processing. This sampled mode of operation actually measures the mean speed over the sample interval and is thus subject to a slight inaccuracy as follows. A typical rotor speed characteristic during a fault condition can be approximated to a sine wave of amplitude 2.0 rad sec^{-1} and period 0.8 sec superimposed on the synchronous speed (for example see Fig 8.19). The effect of averaging the basically smooth monotonic signal over a small interval is shown in Fig 6.1. The mean level of the signal over the interval t_1 to t_2 is available to the processor at time t_2, approximately half a sample interval after the mean level occurred. With a 20 mS sample period and the transient velocity characteristic as described above this is equivalent to a 4^{o} phase shift. This is not considered to be of practical significance.

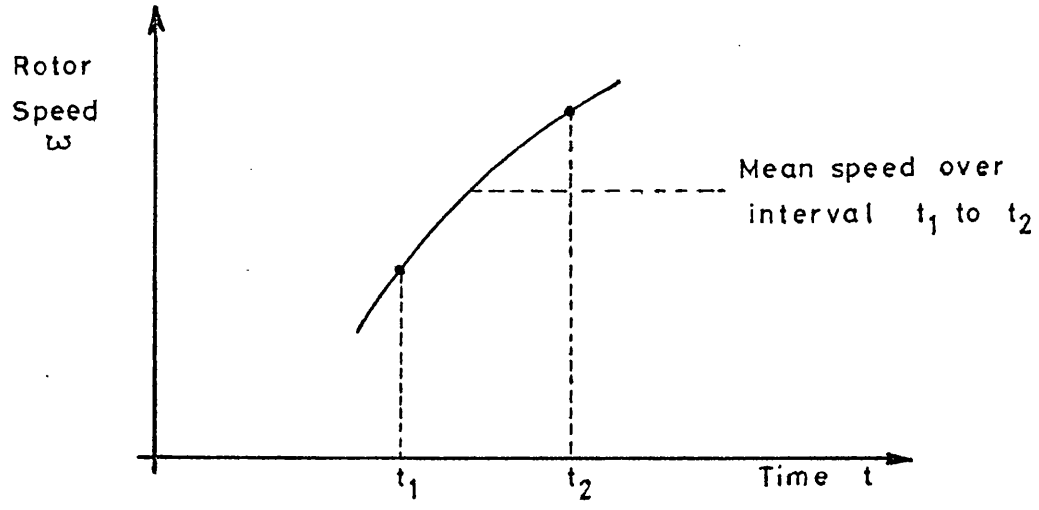


Fig 6.1 Effect of sampling on Transient Velocity

The conversion of the measured period (T) to a frequency is performed by the microprocessor. This process would normally require a division operation to be performed:

$$\omega = \frac{2\pi}{T} \qquad (6.1)$$

Division on the I8080 microprocessor is a relatively slow process, typically 700 μS for 8-bit division and 3 mS for 16-bit division if a hardware multiply/divide facility is not available. To overcome this drawback, eqn (6.1) is rearranged to reflect changes in the speed as a function of changes in the period:

$$\omega + \Delta\omega = \frac{2\pi}{(T+\Delta T)} \qquad (6.2)$$

$$\Delta\omega \simeq -\frac{2\pi\,\Delta T}{T^2} \qquad (6.3)$$

To further simplify the equation, a constant value, T_s, is used in place of the actual period T. This is the synchronous shaft period from which T does not vary by more than 1% giving

$$\Delta\omega \simeq -K_w\,\Delta T \qquad \text{where } K_w = \frac{2\pi}{T_s^2} \qquad (6.4)$$

Thus K_w is a pre-scaled constant which is programmed into the software system making for much faster system operation. The possible error in the transient velocity signal produced by this approximation is only 2% and could be eliminated if actual division in the microprocessor is permitted.

6.2 LOAD ANGLE

The load angle of the synchronous generator is measured by an extended facility of the transient velocity transducer. A complete description of the device is given in the Appendix, Section 8, and only the basic operating features will be discussed here. The load (or rotor) angle of the synchronous machine system is the angle between the quadrature ('q') axis and a synchronously rotating reference, in this case the voltage at the infinite busbar. At any instant in time, the shaft position with respect to an arbitrary fixed position is an angle α which, under steady state conditions, is increasing uniformly from zero to 2π radians every period of shaft rotation. (NB. For simplicity, it is assumed that a two-pole machine is under discussion. For the actual model machine with four poles, one electrical revolution occurs in half a shaft revolution and so a complete shaft revolution is assumed to occur in 180 true mechanical degrees). This characteristic is shown by the 'α' curve in Fig 6.2.

If the synchronous generator is 'floated' on the busbars, that is, with no imported or exported power, then there is a constant phase relationship between the angle α and the voltage at the infinite busbar, as shown in Fig 6.2. If a reading of the angle α is obtained at the same point in each cycle, as indicated by the positive-going zero-crossing of the phase A voltage at the infinite busbar, then a base reference angle, α_o, is obtained. This angle may be stored by the microprocessor

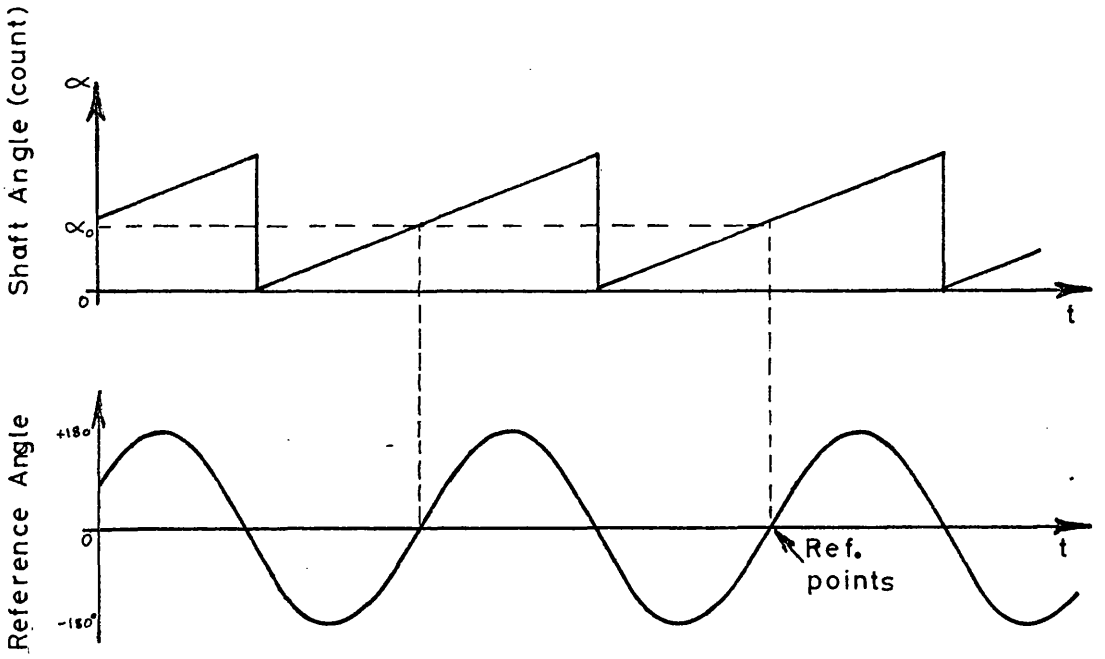


Fig. 6.2 Shaft Angle to Reference Relationship

(no load condition)

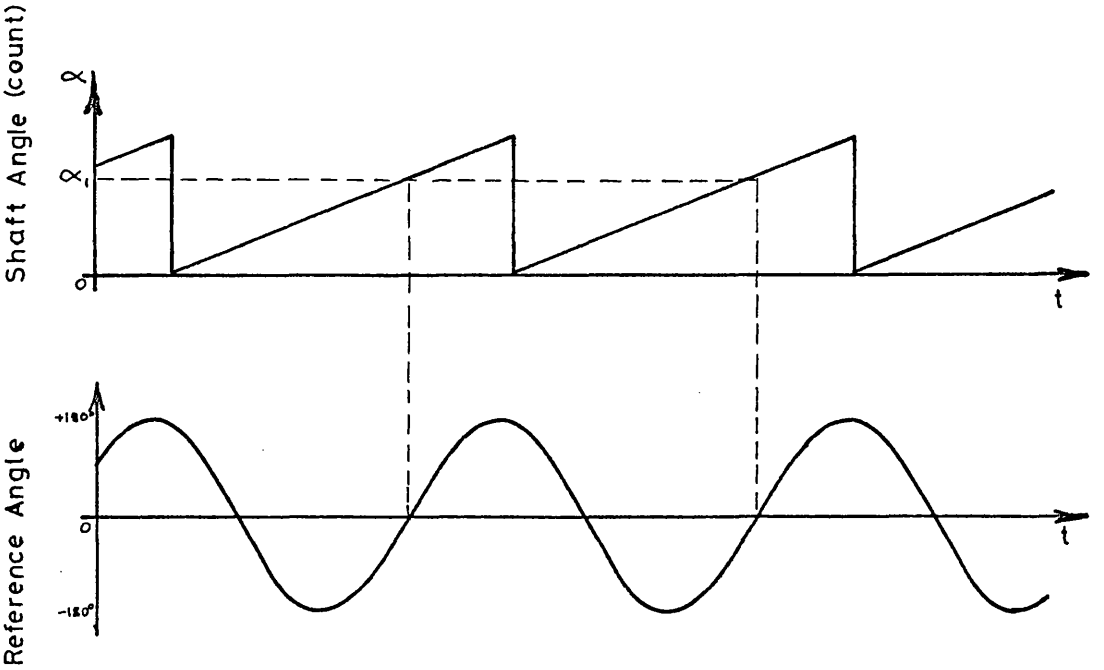


Fig. 6.3 Shaft Angle to Reference Relationship

($\delta \approx 70°$)

as that corresponding to zero load angle.

Under generating conditions, the load angle will be greater and the shaft angle characteristic, α, will advance with respect to the reference as shown in Fig 6.3. This results in a higher reading of the angle α at the zero-crossing point. The load angle is now simply calculated as the difference between the previous and current readings, such that

$$\delta = \alpha_1 - \alpha_o \qquad (6.5)$$

In order for the microprocessor to be able to read the angle α, it is necessary to either have some form of (binary) coded disc on the machine shaft or to use alternative means. It is not considered practicable to mount a coded disc on the shaft with sufficient robustness and accuracy to meet the demands of both the transient velocity and load angle measurements, so an alternative technique has been adopted.

The alternative technique adopted is dependent on the fact that the speed variations of the synchronous machine shaft are not very great. As described in the previous section, the speed characteristic can be approximated to

$$\omega = \omega_s + \omega_d \cos 2\pi f t \qquad (6.6)$$

where ω_s = 314.16 rad sec^{-1}, ω_d = 2.0 rad sec^{-1} and f = 1.25 Hz. Over the small sampling period the speed change is given by

$$\Delta\omega \simeq -\omega_d \frac{(2\pi f.\Delta t)^2}{2} \cos 2\pi ft + 2\pi f.\Delta t \sin 2\pi ft \qquad (6.7)$$

which has a maximum value when $2\pi ft = \pi/2$ giving

$$\Delta\omega_{max} \simeq -\omega_d.2\pi f_o.\Delta t \qquad (6.8)$$

With the values quoted above, the maximum change over a 20 mS sample period is 0.3 rad sec^{-1}. This is 0.1% of the nominal shaft speed. Because the speed is basically constant over the sample interval, the transducer can function as follows. A counter is started when the disk slot passes a photo-pickup and stopped when a positive-going zero-crossing of the phase A voltage of the infinite busbar occurs. Under no-load conditions, this count is read by the microprocessor as N_o and corresponds to the angle α_o described above. When the machine is generating, a higher count, N_1, is obtained, from which the microprocessor calculates the load angle as

$$\delta = 2\pi \frac{N_1 - N_o}{N_T} \qquad (6.9)$$

where N_T is the count representing one complete period of shaft revolution as measured by the transient velocity transducer. This results in a potential accuracy of 0.1% for the load angle transducer but in practice an accuracy of 1.0% is accepted by replacing N_T in eqn (6.9) with a constant count corresponding to synchronous speed and eliminating an actual division by the microprocessor.

In the form described above, the transducer requires a signal from the infinite busbar for reference. In a real power system such a signal is not available but could, however, be replaced by a signal derived from the machine terminals which is maintained in signal form by a phase-locked loop circuit during fault conditions. Such a system would not provide a true value of the load angle under steady state conditions although it would be a rough approximation as the machine reactance is generally much greater than the line reactance. In any case, such information is available to the plant engineer by other means. It would, however, provide all the transient information necessary for implementation of the state feedback controller as described in this study.

6.3 VOLTAGE BEHIND TRANSIENT REACTANCE

The voltage behind transient reactance (e'_q) is obtained with the use of a Park's component resolution circuit operating on the machine voltages and currents. In the present operational system, this resolution is performed by analogue multipliers which function much faster in this application than the equivalent digital multipliers because of the complexity of the transformations.

The voltage e'_q is given by

$$e'_q = v_q + x'_d i_d \qquad (6.10)$$

when resistance is neglected. Its generation requires effective measurement of the quadrature axis voltage, v'_q and the direct axis current, i_d. The quadrature axis voltage is given by

$$v_q = V \cos \delta \qquad (6.11)$$

and can be obtained from a multiplier whose output depends on the magnitude of one signal (V) and the cosine of the angle between this voltage and a reference voltage (cos δ). Such a characteristic is produced in a ring-bridge modulator circuit and the actual multiplier is based on this design as described elsewhere[49]. The direct axis component of the armature current is obtained from a Hall-effect multiplier in a similar manner. The output from the summing junction when both multiplier outputs are added through suitable gain compensators is the q-axis voltage behind transient reactance as given by eqn (6.10). This is converted to a digital value for input to the microprocessor by a 10-bit analogue to digital converter, a technical description being given in the Appendix, Section 9.

6.4 TERMINAL VOLTAGE AND FIELD VOLTAGE

The terminal voltage may be measured in one of two basic ways. If it is desired to perform some form of fault analysis, then measurement of each individual line voltage at specified sample intervals will inevitably be required. This would require the use of three analogue to digital converter units

or an analogue multiplexer with a single converter depending on the speed and accuracy requirements of the conversion. In the present system, however, a measurement of terminal voltage is required only for feedback through the AVR and for data logging purposes. For this purpose, a standard and simple three-phase bridge rectifier and filter circuit is used to give a d.c. representation of the terminal voltage. The filter is a second order low pass type with a double break-point at 80 Hz which gives sufficient attenuation of the 300 Hz ripple, whilst allowing adequate response at lower frequencies. This voltage is converted to digital form for the microprocessor by a 10-bit analogue to digital converter.

The field voltage is driven by the TCR unit (Section 3.4) from the output of a 12-bit digital to analogue converter which is controlled by the microprocessor. Special precautions have been taken in the construction of the converter unit to avoid earth loops and other conditions likely to cause noise problems by the use of optical isolators as described in the Appendix, Section 11. These precautions are necessary when the converter drives the high gain input of the AVR which would otherwise amplify any noise present. In order to allow comparative tests to be made (and logged by the computer) on the difference between the analogue and digital control systems, the field voltage is measured by a 10-bit analogue to digital converter also described in the Appendix.

CHAPTER 7

THE SOFTWARE SYSTEM

A large amount of software (computer programs) has been developed for use in the real-time digital control investigations. Basically, this developed software falls into two areas:

1) Support software, designed to facilitate rapid creation and testing of real-time control programs, and
2) Real-time facilities (monitor calls and subroutines) for use by these programs during the actual control operation.

The direct-loading cross-assembler and on-line debugging technique come into the first category, while the device-handlers and real-time function calls come into the second category.

7.1 'MICRO' CROSS-ASSEMBLER PROGRAM

The problems associated with storage and rapid reloading of microprocessor programs have been discussed in Section 5.4. The MICRO cross-assembler program has been designed to interact closely with the hardware design of the Interface Card, to

extend the facilities available for development of microprocessor programs on the PDP-11/20 computer. The most significant feature of the MICRO cross-assembler, in comparison with other assembler programs, is that it has the ability, if so required, to load programs directly into the microprocessor memory. This loading is performed without the use of any intermediate paper tapes or other 'hard' media.

A typical process of development of a microprocessor control program is shown in Fig 7.1. The program is initially written in a conveniently understandable mnemonic form and stored on any suitable PDP-11/20 peripheral storage unit using the editor program, EDIT[45]. Once loaded, the source program remains resident within the PDP-11/20 system and may be assembled into I8080 microprocessor machine code by the MICRO assembler at any time. As the microprocessor hardware and software interface (ie the Interface Card and Monitor device handlers) has been configured to conform with the general PDP-11 system structure, the assembled machine code may be loaded directly to the microprocessor or stored in the same format for later transmission. Subsequent loading of the file to the microprocessor is performed by the use of simple PDP-11 system-oriented commands. In short, the microprocessor may be treated in the same manner as any other peripheral of the PDP-11 system and data may be transferred to and from the I8080 microprocessor by means of the standard peripheral interchange program, PIP[45]. If the contents of the microprocessor memory are lost for any reason, they may thus be

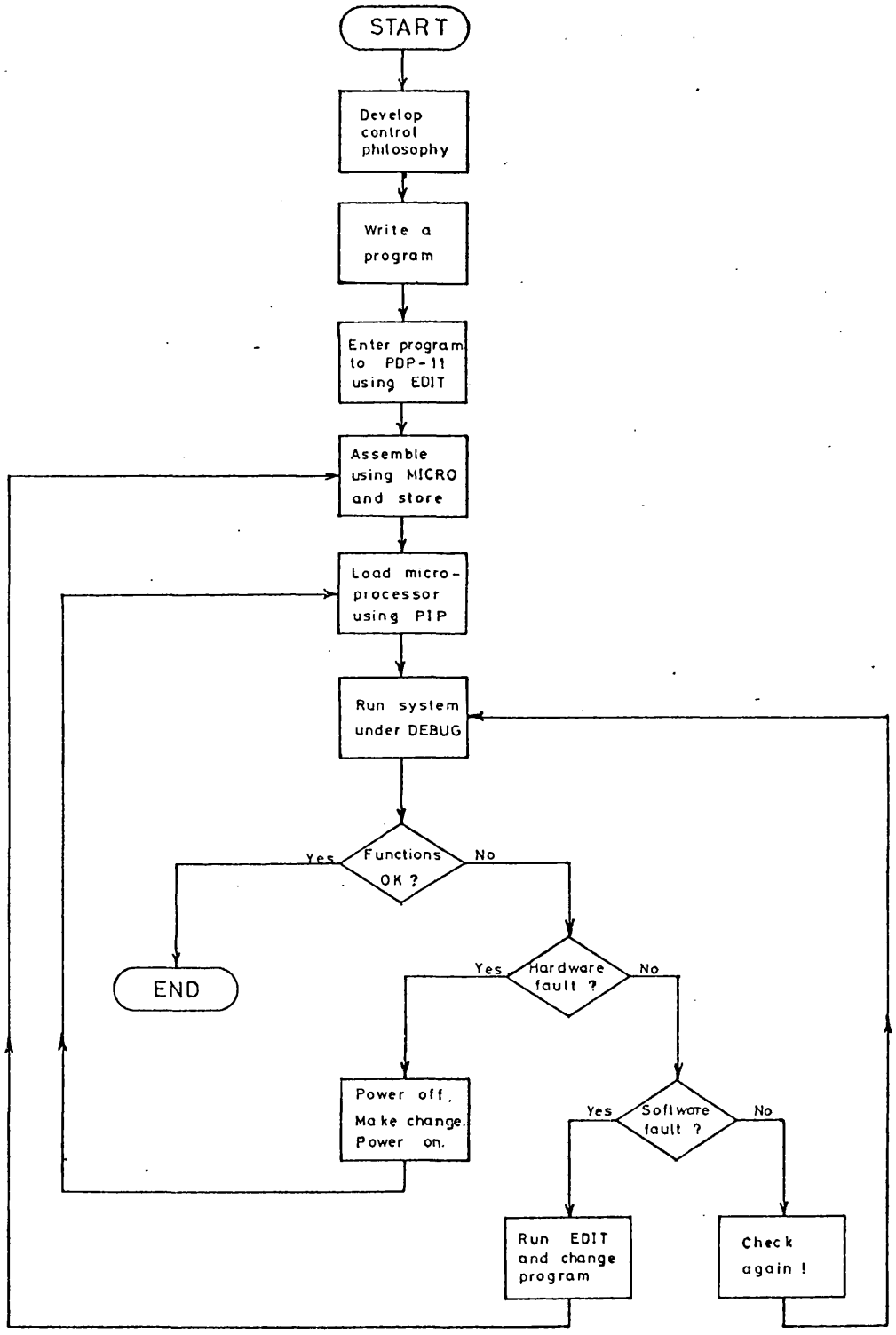


Fig. 7.1 Microprocessor Control Development Sequence

reloaded in the short time taken to type a simple 'PIP' command on the teletype of the supervisory computer. Program changes may be made almost as rapidly by the editing and re-assembly of the stored source program. This facility offers a means of development of microprocessor programs which is many times faster than the use of cross-assemblers using intermediate program transfer media. It also has the advantage over microprocessor-resident monitor and assembler systems in having superior program storage and listing facilities and making the whole of the microprocessor memory available for control program development. A complete description of the MICRO cross-assembler is given in the Appendix, Section 14.

7.2 'DEBUG' ON-LINE DEBUGGING TECHNIQUE

Debugging a microprocessor program can be a very laborious and time consuming process, particularly if the microprocessor system does not have a moderate range of debugging and peripheral support facilities. In the early phases of the research, a 'front panel' controller was used to develop the microprocessor programs. Basically, this was a hardware logic unit which had access to the microprocessor address, status and data busses as well as certain control bus lines. This facility allowed program flow to be followed and simple program changes to be made during the debugging process. The information relating to program operation was relayed back

to the user by means of light-emitting diode arrays. The usefulness of this facility was limited to the development of small programs as the user/microprocessor interaction was totally in machine code (binary) and thus difficult to follow in large sections at the user level. The system also suffered from the drawback that, as the program information was stored in random access memory (RAM) for convenient modification, the program was lost whenever a serious program 'runaway' occurred or a power down operation was required for hardware modifications. Subsequently, it was decided to improve the facilities available by designing the microprocessor/computer interface, already required for data transmission, such that program evaluation and debugging could be performed under the control of the PDP-11. Such a facility is provided jointly by the hardware of the Interface Card and the software of the on-line debugging technique, DEBUG.

Use of DEBUG, described in the Appendix, Section 15, enables the user to test a microprocessor program on an actual processor in a real-time environment with the high level software support facilities of a large computer. This is considered to be a significant improvement over the use of 'emulator' programs run on a large host computer, as the majority of problems in real-time programming have been found to be in interfacing with the transducers and other peripherals. Such problems can only be thoroughly investigated on the actual real-time system. Using DEBUG, the user can optionally examine and change memory locations, run or single-step

programs, simulate interrupts or data-transfers and perform other operations to aid the debugging process. In performing this process, the user is aided by the fact that all commands are issued through the convenient facility of the main computer teletype.

7.3 REAL-TIME MONITOR SYSTEM

The monitor system used on the PDP-11/20 computer is a slightly modified form of the Digital Equipment RT-11 system[45,48]. This monitor system has been modified to include device handlers for loading programs to the microprocessor and for transferring data to and from the microprocessor during the real-time control situation. With these modifications, the microprocessor effectively becomes another peripheral of the PDP-11 system and advantage may be taken of the full flexibility of device independent operation. For example, to develop a data-logging program on the PDP-11, using data from the microprocessor, the following procedure may be adopted: one typical set of sampled data may be transferred from the microprocessor to the magtape, say, using the standard peripheral interchange program, PIP. The data logging program may then be debugged, using the magtape unit as the input device. This avoids the necessity of the continual running of the system under test, in this case the synchronous generator system. Once the program is successfully debugged, the input device may be re-allocated by a simple monitor command to put the test system on-line. Several

other useful features are exhibited by the RT-11 monitor system. It has a foreground/background programming facility, enabling programs to be developed in the background mode while performing a control function in the foregound mode. This facility would also be useful in a power station environment, allowing data logging tasks and modelling functions to be performed in the background mode, reserving the high priority foreground mode for immediate control during system disturbance conditions. A further feature of the RT-11 system monitor is that it supports an extended real-time version of FORTRAN IV, enabling easy supervisory program development in the present system.

7.4 THE CONTROL PROGRAM

The complete operation of the real-time control program is complex and only the salient features will be discussed here. Further details are given in the relevant section of the Appendix.

7.4.1 Calculation of State Variables

The microprocessor obtains the various states of the system from the transducers as described in Chapter 6. However, the values obtained from the immediate processing of the data provided by each transducer are generally the absolute values of the states. In the theoretical analysis, only the difference of each state from the steady state operating point have been used

to calculate the gains of the feedback matrix and thus some correction must be made. The differences are extracted by means of a simple filtering algorithm performed within the microprocessor. This will be discussed briefly below as it is representative of the modelling technique employed within the microprocessor.

Consider the first order high pass filter as shown in Fig 7.2.

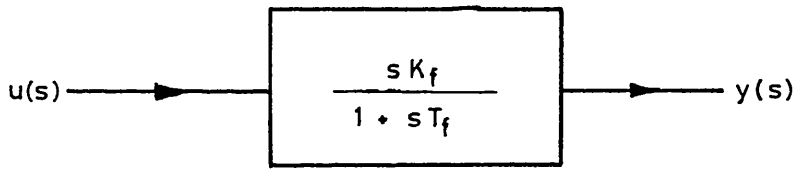


Fig 7.2 Simple High Pass Filter

The microprocessor routine to simulate the filter function is developed as follows. In state space form, Fig 7.2 can be represented as

$$\dot{x}(t) = -\frac{1}{T_f} x(t) + \frac{K_f}{T_f} u(t) \qquad (7.1)$$

$$y(t) = \dot{x}(t) \qquad (7.2)$$

Integrating eqn (7.1) gives

$$x(t) = e^{-t/T_f} x(0) + \int_0^t e^{-(t-\tau)/T_f} \frac{K_f}{T_f} u(\tau) \cdot d\tau \quad (7.3)$$

or, during the sample period $0 \leq t \leq T$

$$x(T) = e^{-T/T_f} x(0) + \int_0^T e^{-(T-\tau)/T_f} \frac{K_f}{T_f} u(\tau) \cdot d\tau \quad (7.4)$$

The processor effectively functions as a sample and hold circuit, such that $u(\tau)$ is constant over the sample interval, ie $u(\tau) = u(0)$.

Then

$$x(T) = e^{-T/T_f} x(0) + \frac{K_f}{T_f} u(0) \int_0^T e^{-(T-\tau)/T_f} \cdot d\tau \quad (7.5)$$

Solving gives

$$x(T) = e^{-T/T_f} x(0) + K_f(1 - e^{-T/T_f}) u(0) \quad (7.6)$$

Eqn (7.6) predicts the state at the end of the first sample instant. Similarly for the second interval:

$$x(2T) = e^{-T/T_f} x(T) + K_f(1 - e^{-T/T_f}) u(T)$$

or, more generally,

$$x(\overline{k+1}\ T) = e^{-T/T_f} x(kT) + K_f(1 - e^{-T/T_f}) u(kT) \quad (7.7)$$

and from eqns (7.1) and (7.2)

$$y(\overline{k+1}\ T) = -\frac{1}{T_f} x(\overline{k+1}\ T) + \frac{K_f}{T_f} u(\overline{k+1}\ T) \qquad (7.8)$$

Eqns (7.7) and (7.8) are the difference equations on which the microprocessor model is based. The actual values chosen for the gain K_f/T_f, and the time constant, T_f, are unity and 5.11 seconds, respectively. The value of time constant allows the controller to follow changes in operating points of the system, yet does not suppress any transients occurring during fault conditions. The exact value has been chosen with a specific purpose to give a value for $e^{-T/T_f} = 2^{-8}$ (as T = 0.02 seconds). This means that the multiplication to be performed within the microprocessor is merely a byte shift operation, significantly improving the processor speed.

The calculations of the feedback function are performed by the processor using the offset state values obtained as above, the functions being defined in eqns (4.2) and (4.3).

7.4.2 Data Logging

While the control system is on line, the PDP-11/20 computer maintains in its core memory a 'rolling stack' of the major system variables received from the microprocessor during the preceding 0.2 seconds of real time. At the detection of a major system disturbance, conveniently chosen in the laboratory model to be a 20% drop in terminal voltage, the PDP-11/20 halts

the rolling action and commences to record the major system variables. This action continues for 9.8 seconds, at the end of which time a request is made to the operator as to whether these results should be permanently stored for immediate or later plotting as desired. This facility is described in detail in the Appendix, Section 18.

Obviously, the facility described above has been specifically tailored to the needs of a laboratory development system. However, minor modifications would provide the sort of data logging facilities required by full-size plant controllers.

7.4.3 System Errors

The digital control scheme described in this thesis is basically a laboratory development version. However, several system checking and fail-safe features have been incorporated, primarily to prevent damage to the expensive micromachine model in the event of a system failure. Further, the experience gained in development of a system in which certain error conditions are considered is useful for the development of digital control systems capable of operation in a practical environment.

All transmissions between the microprocessor and PDP-11 computer are handled by device drivers. On transmission, each block of data is preceded by a block count and terminated by a checksum byte, generated automatically by the device driver. In the receiving system, the device driver forms a checksum which is compared to the one received. In this manner, any

single bit errors are detected by the receiving processor. This system also detects most multiple bit errors as the probability of a multiple bit error with a correct checksum occurring is extremely small if the basic error rate is low. The action to be taken by the respective processors on detection of an error is dependent on the operating environment. In a practical control system in normal operation, it is likely that the error would be flagged and a re-transmission requested. Alternatively, the receiving processor could be programmed to reject the current transmission and base the control action on an estimated value obtained from previous data. Obviously, a high error rate signifies a serious system fault. In the laboratory model, the error is flagged and further transmissions are halted in order to allow investigations of the causatory factors to be made.

The microprocessor is also programmed to perform various 'time-out' checks both on the peripheral transducers and the main PDP-11 computer. If a transducer fails to return a valid reading within its allotted time period, an error message is sent to the PDP-11 computer which notifies the operator of this occurrence. A display is also provided locally on the display panel (see the Appendix, Section 5). The microprocessor then performs a default routine offering 'safe' control as described below. In the event of the PDP-11 computer not responding within an allotted period, the same routine is also adopted as a default process and the error flagged. The general sequence of processing events expected by the micro-

processor is summarised by Fig 7.3 and any different sequence indicates a system error. In the laboratory system, the default control is merely to eliminate the state feedback control and revert to AVR control alone. However, in a more sophisticated system it is envisaged that some attempt to correct the error could be made. This is generally provided by switching to a backup system which is a duplicate of the first.

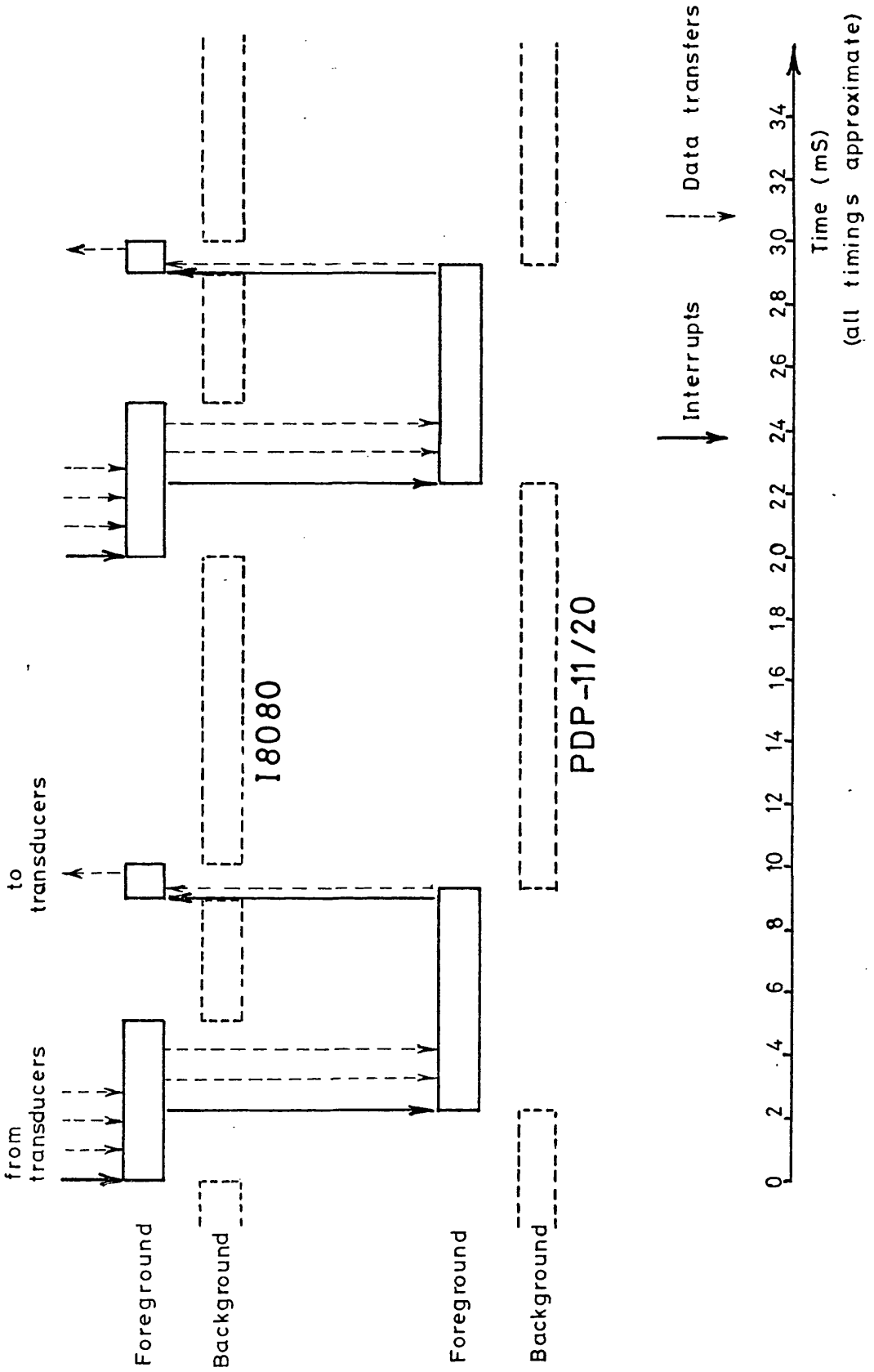


Fig. 7.1 Real Time Control Program Timing Diagram (Simplified)

CHAPTER 8

EXPERIMENTAL RESULTS

This chapter describes the experimental results obtained from tests on the micro-machine system using various control laws of the form discussed in Chapters 2 and 4. For historical reasons, these results may be classified into three main areas generally following the development of the digital control system to its current state. These regions are broadly classified as: comparison of analogue-derived state feedback to digital state feedback, control by digital state feedback into the low-gain point of the AVR and control by digital state feedback into the high-gain point of the AVR.

8.1 COMPARATIVE TESTS OF ANALOGUE TO DIGITAL STATE FEEDBACK

Prior to the introduction of the digital control system, attempts were made to control the micromachine model by feedback of a linear combination of certain states which had been generated by conventional analogue techniques. As discussed in previous chapters, the three states considered for feedback were:-

δ	load angle (or rotor angle)
$p\delta$	rotor transient velocity
e'_q	q-axis voltage behind transient reactance

The performance of various types of transducer was investigated prior to the current study[47,34] and the types discussed below were found to give best results. The performance criteria were accuracy and freedom from noise, the latter being more difficult to achieve.

8.1.1 Analogue Transducers

Generation of a voltage signal proportional to load angle was achieved with the use of a small three-phase a.c. tachogenerator. This tachogenerator was mechanically tightly coupled to the micro-alternator shaft such that its output voltage was in synchronism with the open circuit output of the micro-alternator. This produced an output from the tachogenerator which was always in phase with the shaft rotation. An electronic phase comparator circuit was then used to generate a voltage proportional to the phase difference between the tachogenerator output and the infinite busbar voltage. By the conventional definition, this signal was proportional to the rotor angle.

The rotor transient velocity signal was produced by passing the rotor angle signal through a pair of 50 Hz and 100 Hz notch filters to remove the major noise components and then differentiating the signal electronically. The method described in Section 6.3 was used to generate the q-axis voltage behind transient reactance signal and all three signals were then summed by conventional operational amplifier current-summing techniques. The voltage signal thus produced was 'backed-off'

against a variable voltage level to remove the steady-state components of the feedback signal.

8.1.2 Analogue State Feedback to High Gain Point of the AVR

Attempts were made to implement an analogue controller in the form of Fig 4.2 with the additional feedback signals summed into the high gain input of the AVR, corresponding to the configuration available in practice. However, the presence of the high gain in the forward path of the AVR led to a high amplification of the noise components present in the state feedback signals. This effect resulted in the introduction of large oscillations in the voltage applied to the field of the micro-alternator as soon as the extra state signals were mixed in. As a direct consequence, the system became unstable before the application of a system fault and it has not been possible to perform any representative tests with this feedback arrangement.

8.1.3 State Feedback to the Low Gain Point of the AVR

As the presence of noise prevented the addition of the state feedback signal to the input summing junction of the AVR, it was decided to test the effectiveness of state feedback in performance improvement by adding the signal after the high forward gain of the AVR as shown in Fig 4.1. Although this technique would not be possible in practice (the exciter

output being at intermediate power levels), it would demonstrate the effectiveness of the state feedback system in improving the performance of the model system prior to further investigations.

The state feedback parameters were optimised to minimise the performance index of the configuration as detailed in Chapters 2 and 4. Using the system parameters of Tables 3.1 and 4.1 the sub-optimal feedback law was found to be:

$$V_i = -20.0\ e'_q - 5.9\delta + 2.5\ p\delta \qquad (8.1)$$

when operating at a power of 0.85 pu with the machine terminal voltage set to 1.0 pu and with unity power factor at the infinite busbar. The standard fault condition corresponded to a symmetrical three-phase short circuit, of duration 140 mS, at the high voltage terminals of the generator transformer with the pre-fault impedance equal to the post-fault impedance.

Figs 8.1, 8.2 and 8.3 show, respectively, the excitation voltage, the terminal voltage and the load angle of the micro-machine model system when a fault was applied at the above conditions.

The three forms of feedback used were: 1. The AVR of Fig 4.1 with no state feedback ($V_i = 0$), 2. The AVR plus the state feedback of equation (8.1) derived by analogue means, and, 3. the AVR plus the same state feedback law derived by digital means.

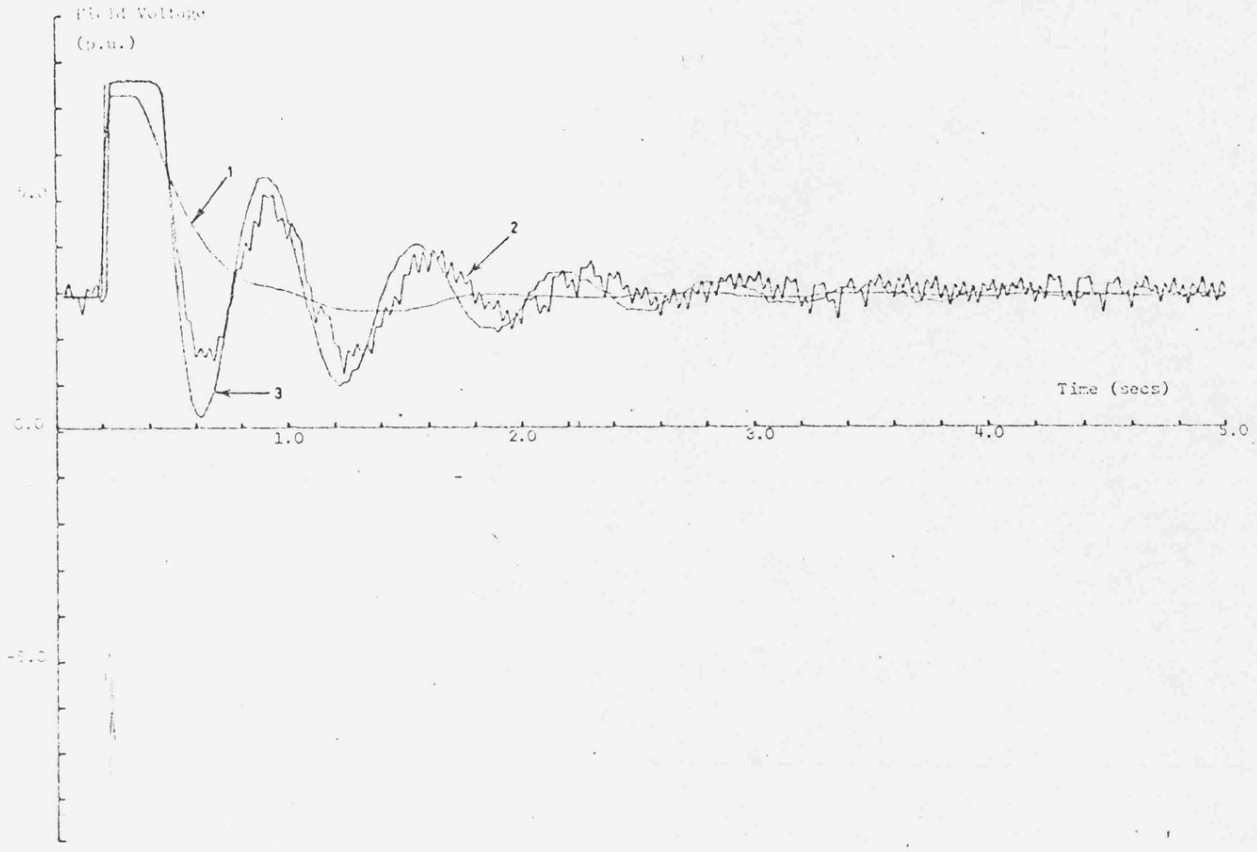


Fig. 8.1 Field Voltage (Excitation) against Time

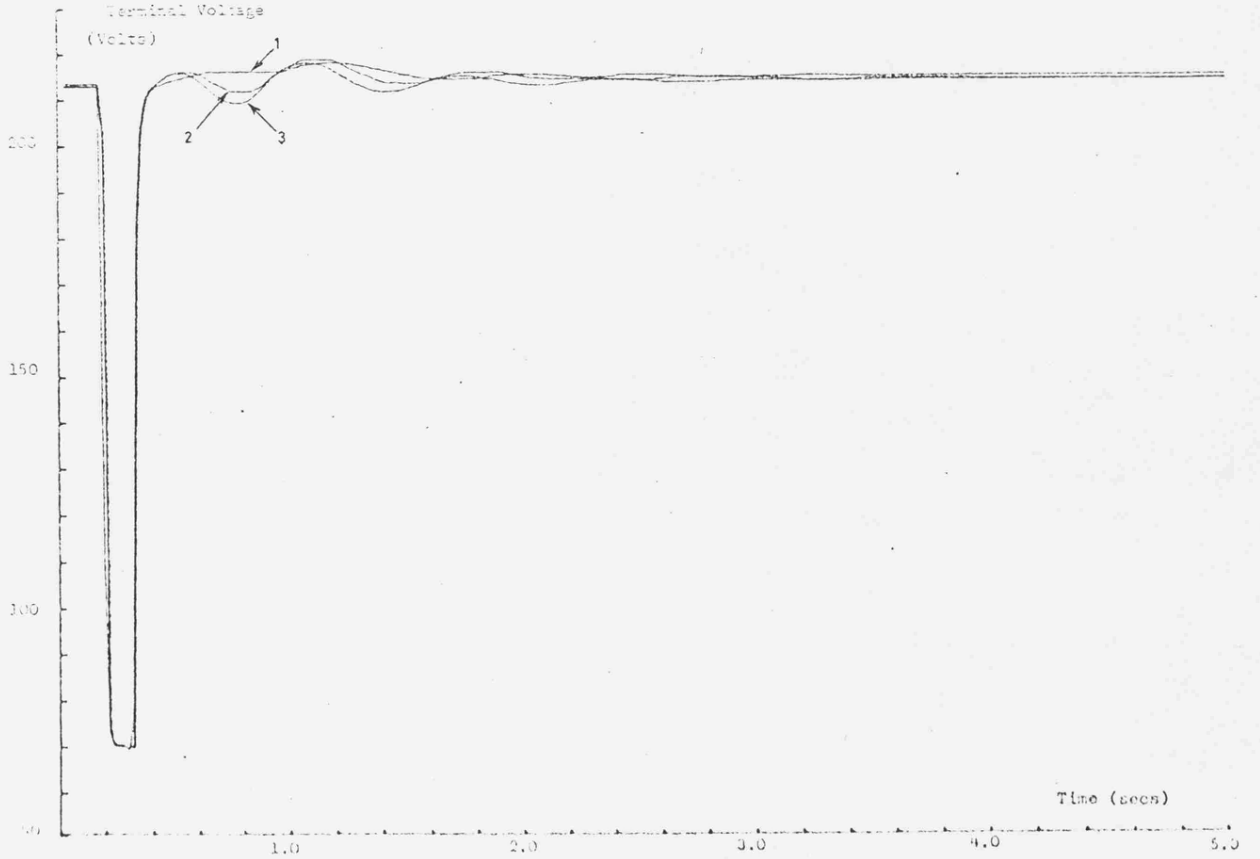


Fig. 8.2 Terminal Voltage against Time

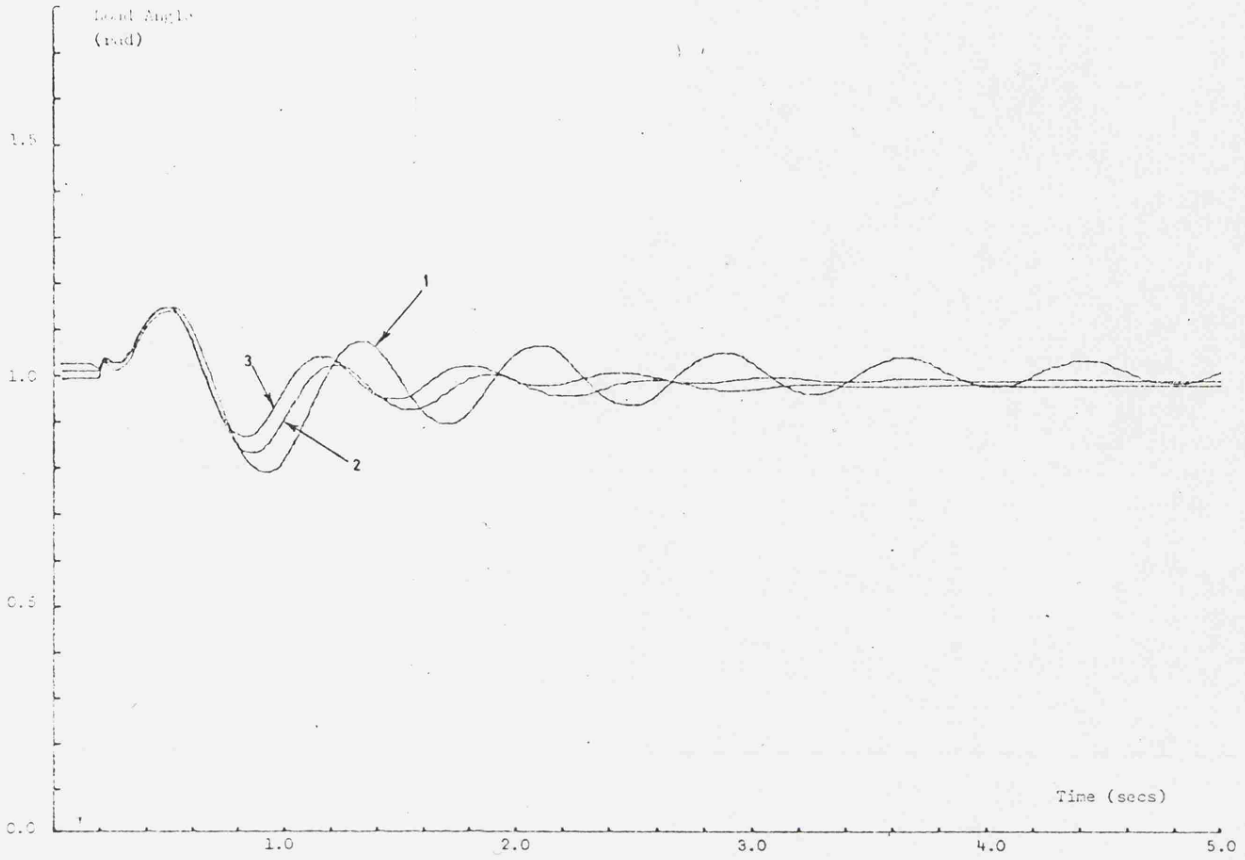


Fig. 8.3 Load Angle against Time

DETAILS OF FIGS. 8.1, 8.2 & 8.3

Configuration: Low gain (as Fig. 4.1)

Plot No. 1: AVR Control ($V_i = 0$)

Plot No. 2: AVR plus Analogue State Feedback (V_i as eqn. 8.1)

Plot No. 3: AVR plus Digital State Feedback (V_i as eqn. 8.1)

Operating Point: D (Standard — Table 8.1)

Fault Duration: 140 mS

The reason for abandoning the implementation of state feedback by analogue techniques becomes clear by examination of Fig 8.1. These curves show the excitation applied to the machine field during and immediately following the application of the fault. The two forms of state feedback controller have similar characteristics and give similar performance but the presence of an unacceptable noise component in the analogue-derived law is easily seen. It is this component which causes instability in the only practically viable situation of state feedback into the summing (high-gain) input of the AVR. However, both forms of state feedback can be seen to produce a similar improvement in the system response which will be discussed in respect of the digital control system in subsequent sections. Further work on analogue state feedback systems was abandoned in favour of digital systems.

8.2 DIGITAL STATE FEEDBACK (TO LOW-GAIN POINT OF AVR)

As the model system had been configured for tests with analogue state feedback to the low-gain point of the AVR, early tests on the digital state feedback system were continued in this configuration to determine the effectiveness of the control while system restructuring and parameter reoptimising was taking place. These tests also allowed some measure of the sensitivity of the practical system to the feedback parameters for different operating conditions to be made. This was performed in a qualitative manner by using a feedback control law

which had been optimised for one set of conditions as a control for a different set of conditions.

8.2.1 Operating Conditions

Throughout the rest of the study, tests were performed on the system at a matrix of operating points and conditions. These involved a variation of the steady state excitation to produce leading and lagging volt-ampere reactive regions and variations in fault duration. These operating points and conditions are summarised by Fig 8.4 and Table 8.1.

8.2.2 Unity Power Factor Operation

Short circuit tests were performed with the micromachine system operating at point 'D' (Fig 8.4), unity power factor at the infinite busbar. The state feedback gains of eqn (8.1) are the result of an off-line optimisation at this point for a fault duration of 140 mS and so this case was investigated first.

8.2.2.1 Standard Fault Duration

Figures 8.5 to 8.7 show the results of a test involving the application of a 140 mS fault to the high voltage terminals of the generator transformer. Three forms of control were used for comparative purposes:

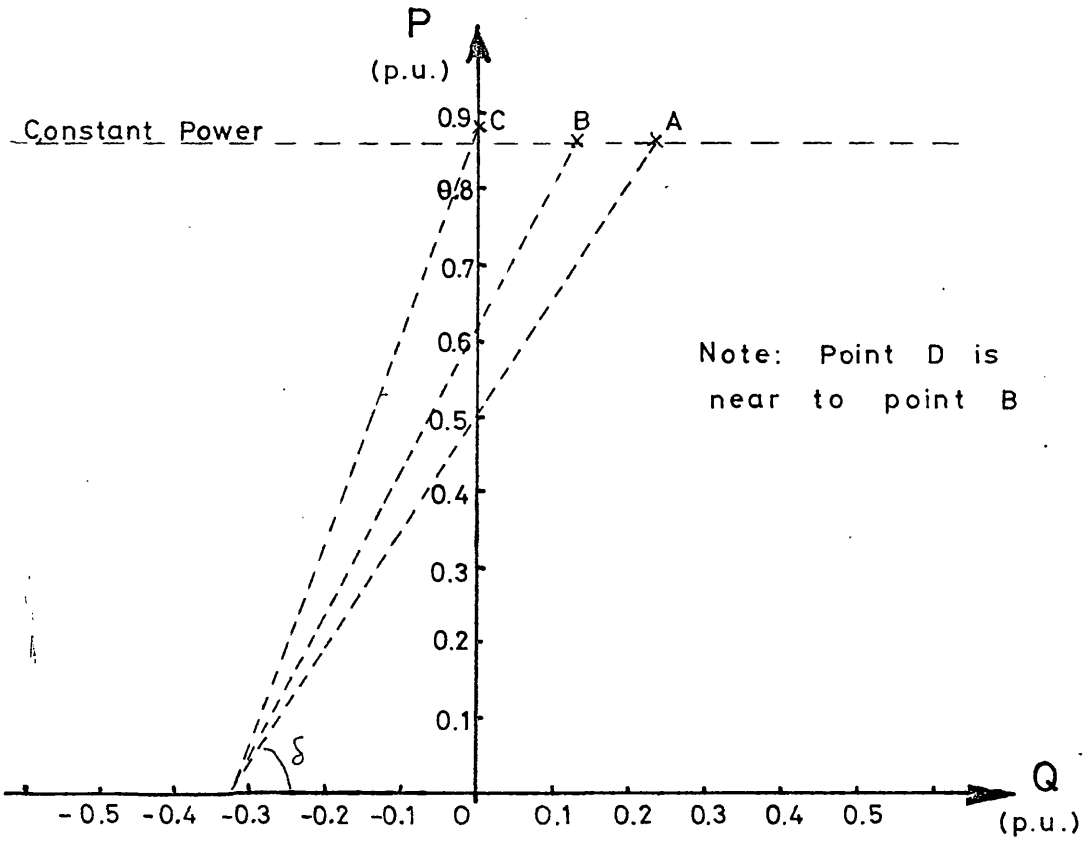


Fig. 8.4 Sketch of Operating Points during Tests

Test Point	Power KWATT p.u.		Reactive Power KVAR p.u.		Term Voltage (volts)	Busbar Voltage (volts)	Load Angle deg	rad	Field Voltage (p.u.)
A	3.00	0.869	0.81	0.235	206	187.5	58°	1.02	3.19
B	3.00	0.869	0.45	0.13	206	196.5	65°	1.14	2.85
C	3.04	0.88	0.0	0.0	206	209.0	76°	1.32	2.49
D	2.97	0.861	0.62	0.18	212	196.0	60°	1.05	2.86

Table 8.1 Model Power System Operating Points

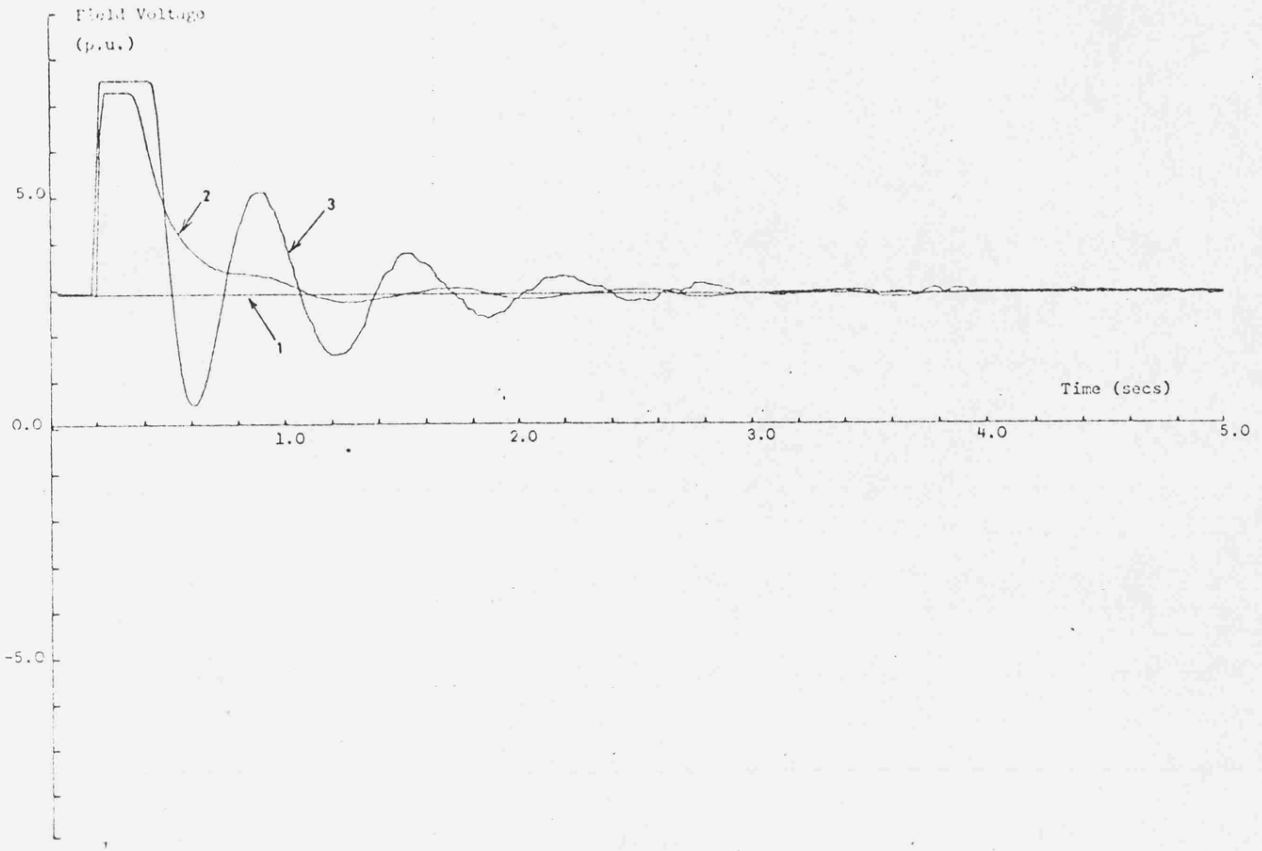


Fig. 8.5 Field Voltage (Excitation) against Time

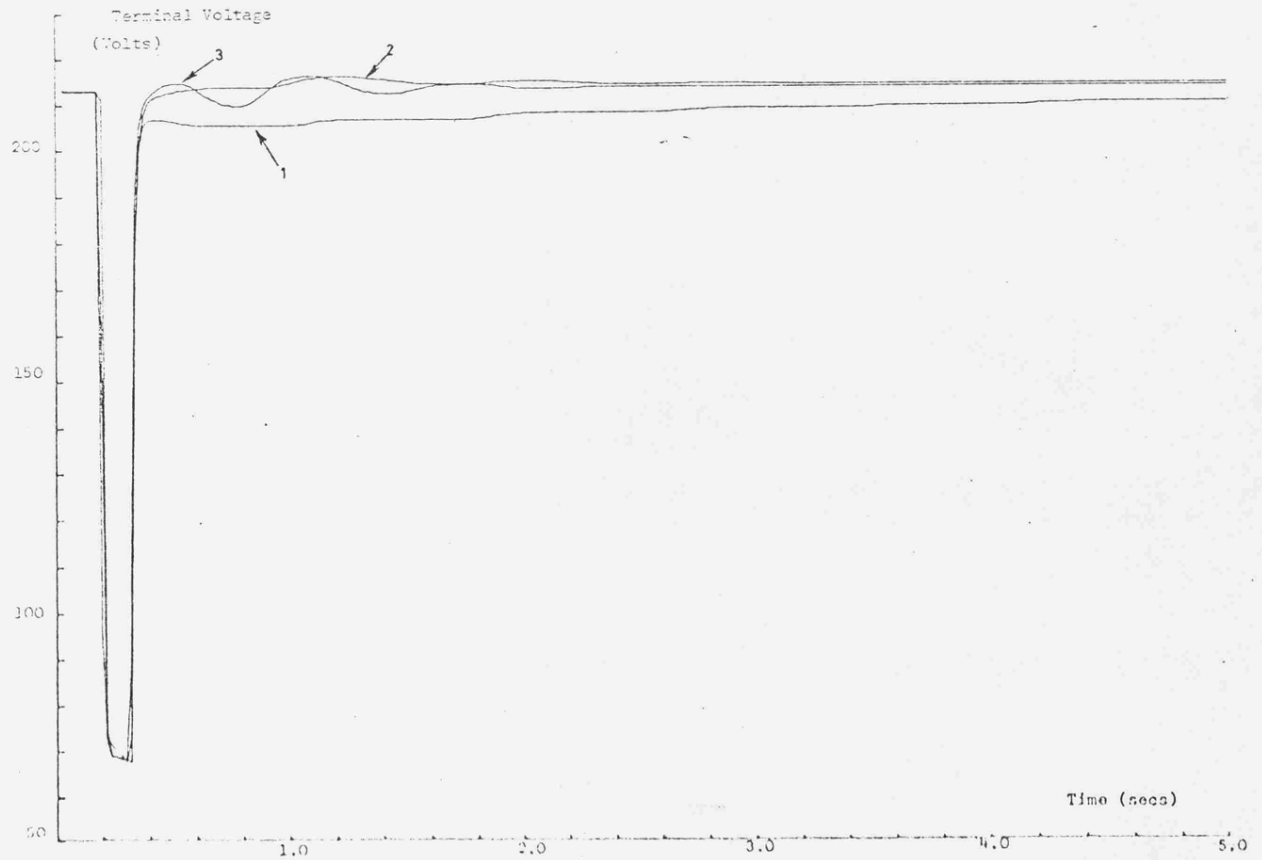


Fig. 8.6 Terminal Voltage against Time

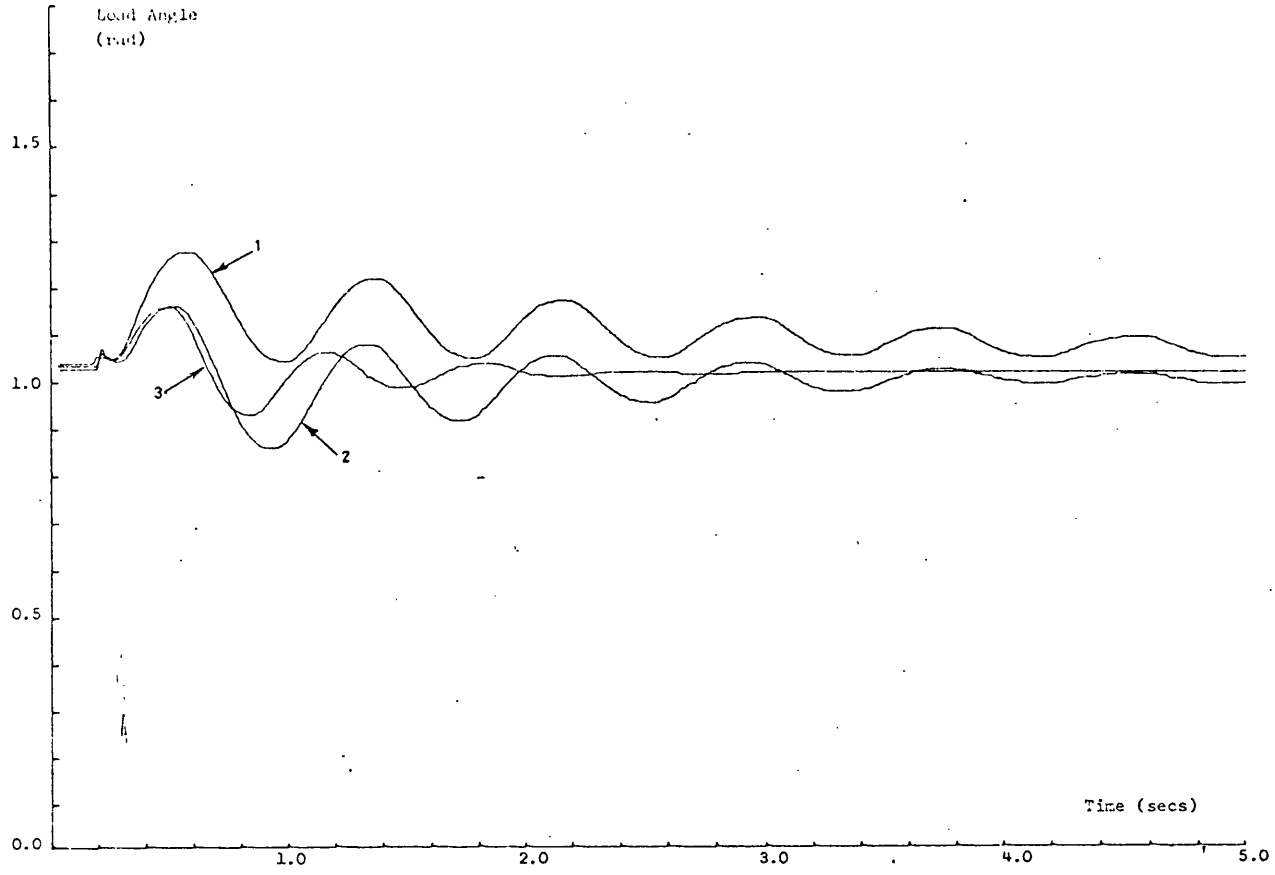


Fig. 8,7 Load Angle against Time

DETAILS OF FIGS. 8.5, 8.6 & 8.7

Configuration: Low gain (as Fig. 4.1)

Plot No. 1: Constant Excitation

Plot No. 2: AVR Control ($V_i = 0$)

Plot No. 3: AVR plus Digital State Feedback (V_i as eqn. 8.1)

Operating Point: D (Standard — Table 8.1)

Fault Duration: 140 mS

1. Constant excitation

2. The AVR without any additional control signals

3. The AVR with the state feedback of eqn (8.1) at the low-gain point

Examination of Fig 8.6 shows that the recovery of terminal voltage following the fault removal was equally rapid for both the AVR and state-feedback controllers. This was expected due to the application of maximum positive excitation in both cases following a fall in terminal voltage. Naturally, the system driven by constant excitation takes a long time to recover. Further, the fact that maximum positive excitation was applied by both types of feedback controller during the first forward swing of the rotor meant that there was essentially no difference between the AVR and state feedback controls in limiting the peak of this first swing. This effect is shown by the rotor angle curves of Fig 8.7. At this point, both controllers have reduced the first peak swing to approximately one half of that occurring under constant excitation conditions. However, subsequent swings were significantly reduced in amplitude and more rapidly suppressed by the presence of the state feedback signals in the control law indicating a system which is inherently more stable.

8.2.2.2 Other fault durations

Tests were also performed at the same operating point but with fault durations of 80 and 220 mS. The results are given in Figs 8.8 to 8.10 and 8.11 to 8.13 respectively. Similar characteristics are shown by the tests involving an 80 mS fault to those previously discussed for a 140 mS fault, indicating that the state feedback controller produced a marked improvement in performance with respect to the rotor angle swings. However, in the case of the 220 mS duration fault, the initial recovery of the terminal voltage was as rapid for both types of feedback control, as shown in Fig 8.12, but the subsequent action applied by the state feedback controller to reduce the rotor angle swings was detrimental to the terminal voltage behaviour. This indicated that the control law was possibly no longer the desired sub-optimal one under these different conditions and would need re-optimising with a possible change in weighting factors in eqn (4.1) to produce the desired response. This line was not pursued as this form of control is not practicable to implement for the reasons discussed in Section 8.1.3 and further investigations were continued with the feedback applied to the high-gain point of the AVR.

8.3 DIGITAL STATE FEEDBACK (TO HIGH-GAIN POINT OF AVR)

The original concept of a practical state feedback scheme for a turbogenerator system was such that the control computer

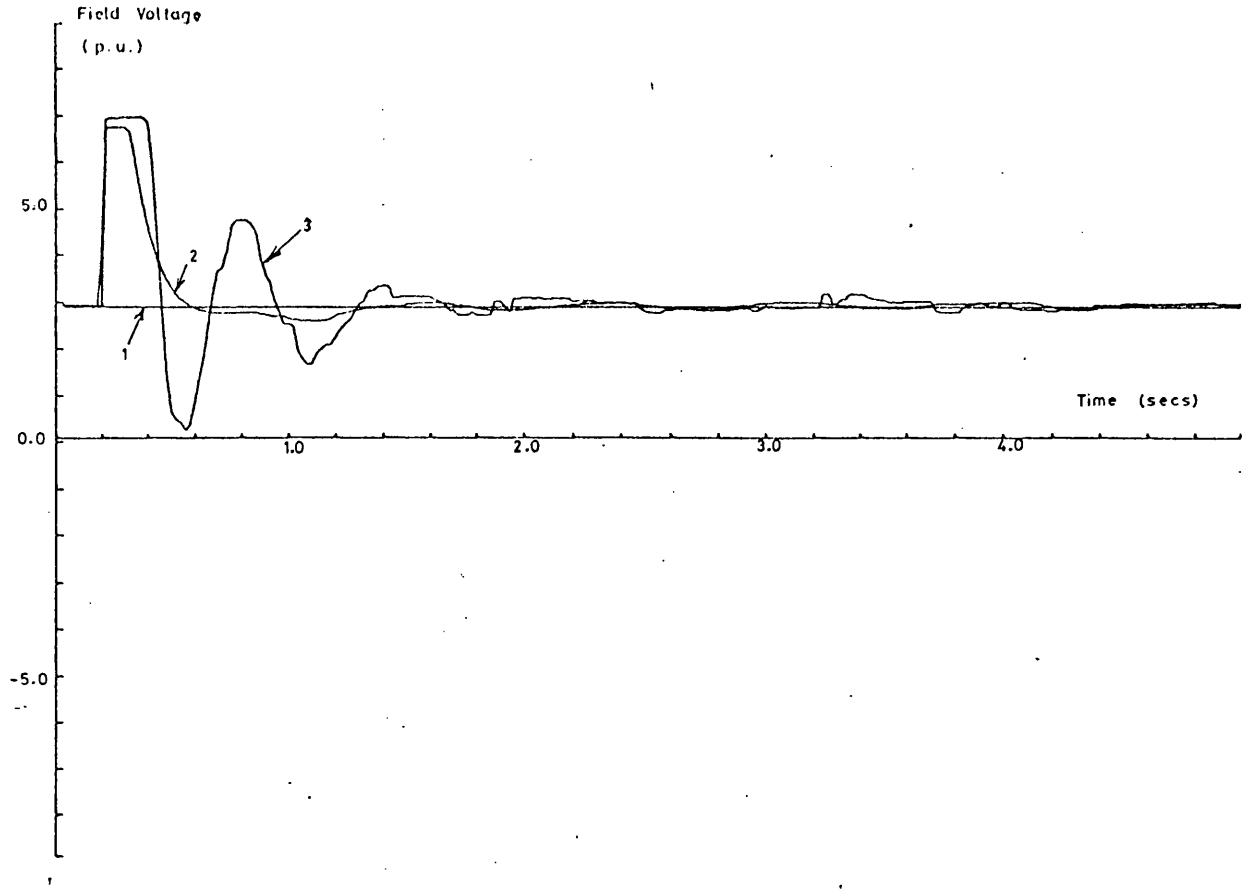


Fig. 8.8 Field Voltage (Excitation) against Time

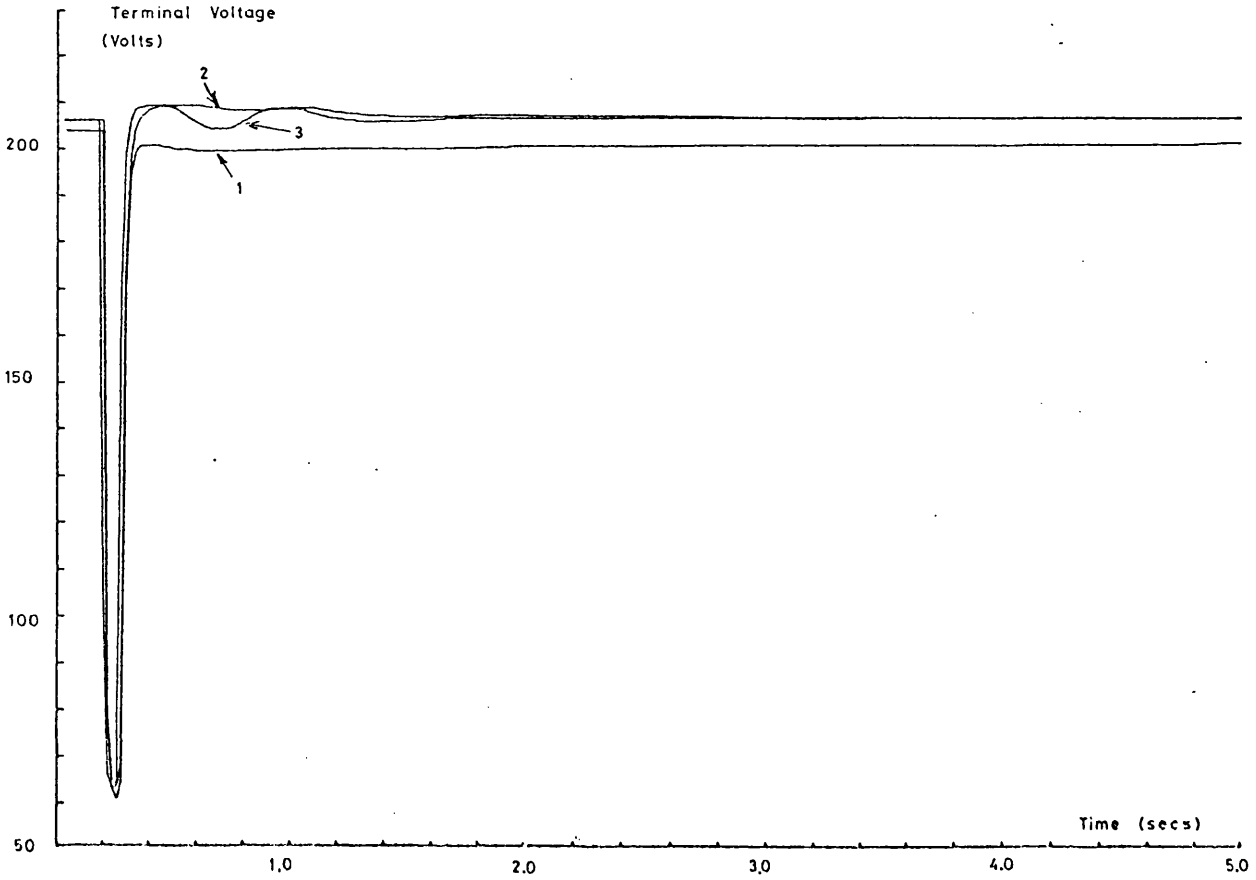


Fig. 8.9 Terminal Voltage against Time

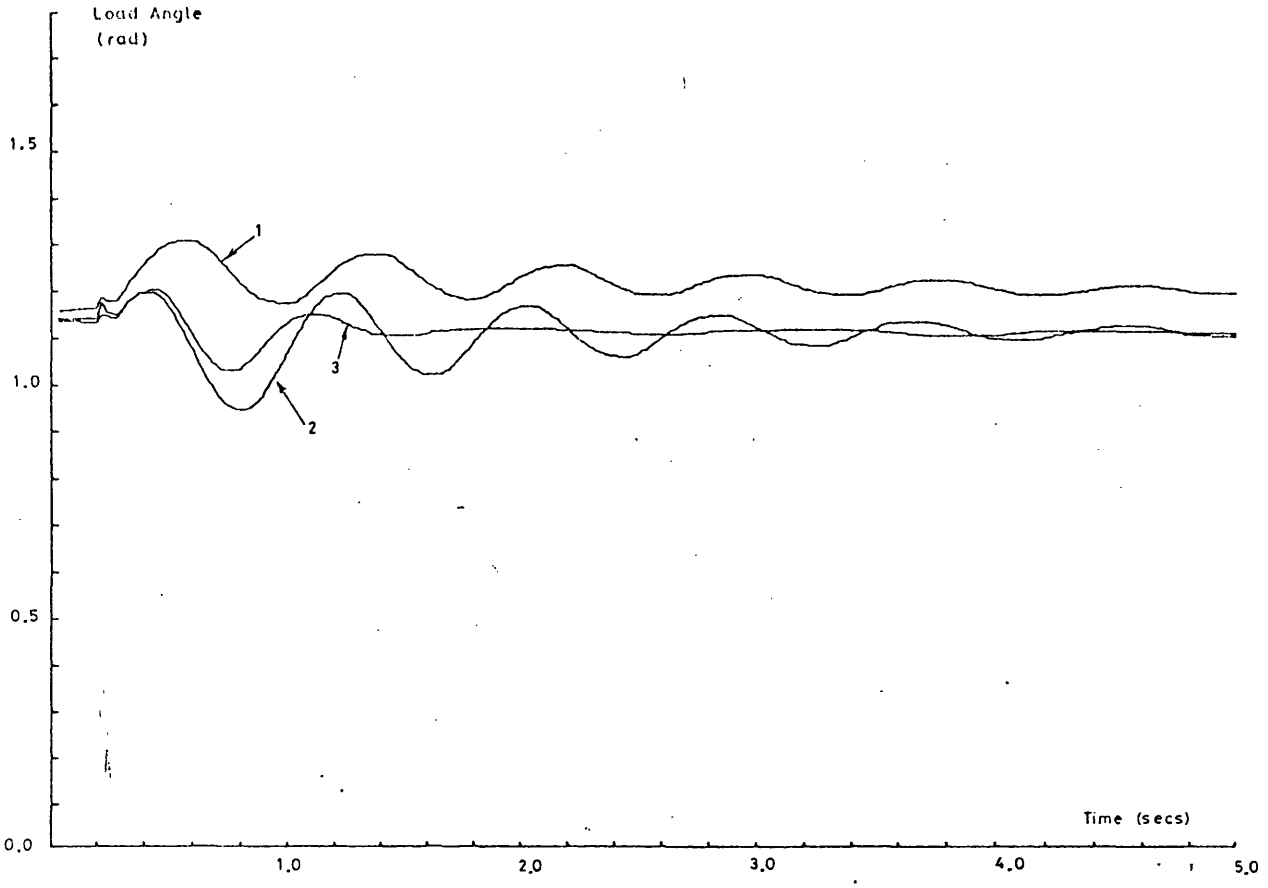


Fig. 8.10 Load Angle against Time

DETAILS OF FIGS. 8.8, 8.9 & 8.10

Configuration : Low gain (as Fig. 4.1)

Plot No. 1 : Constant Excitation

Plot No. 2 : AVR Control ($V_i = 0$)

Plot No. 3 : AVR plus Digital State Feedback (V_i as eqn. 8.1)

Operating Point : B (Standard — Table 8.1)

Fault Duration : 80 mS

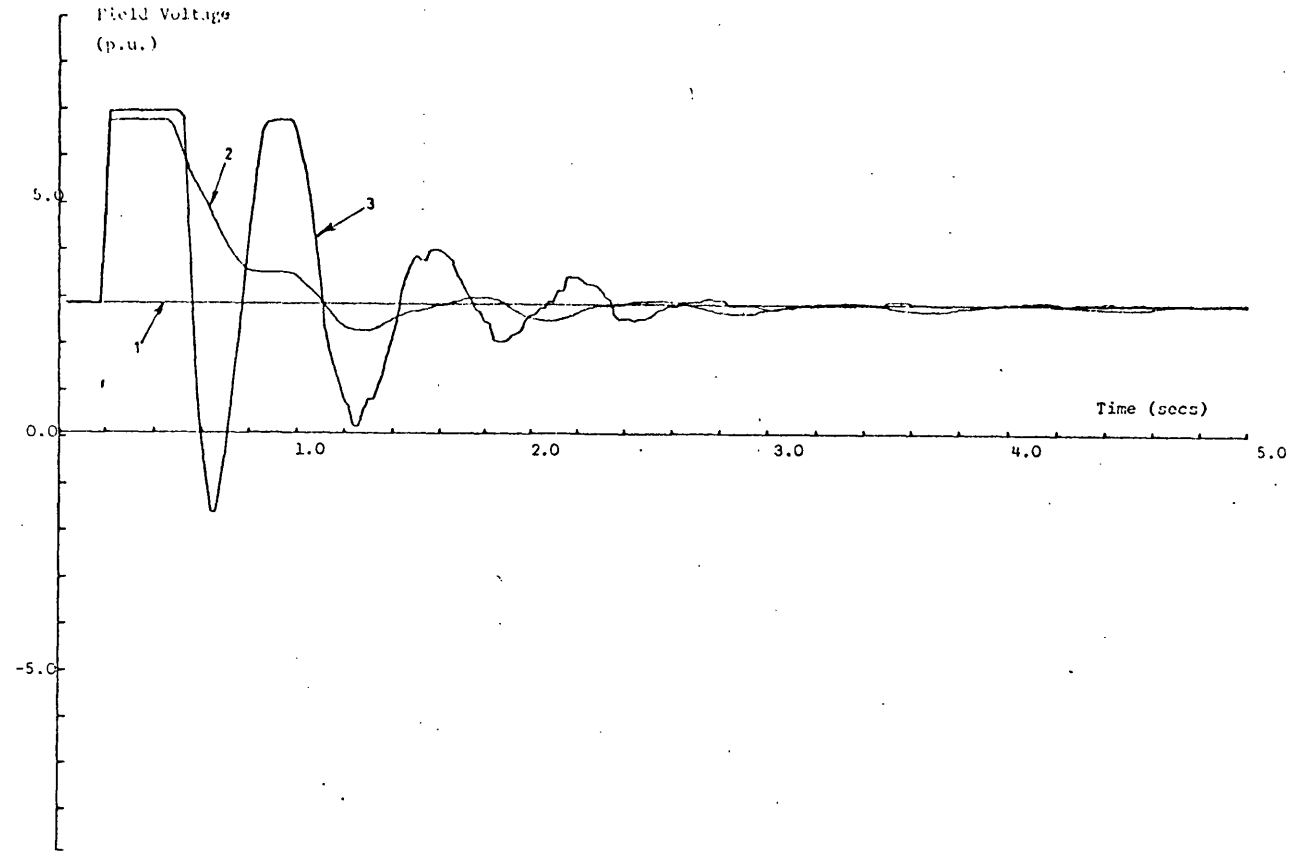


Fig. 8.11 Field Voltage (Excitation) against Time

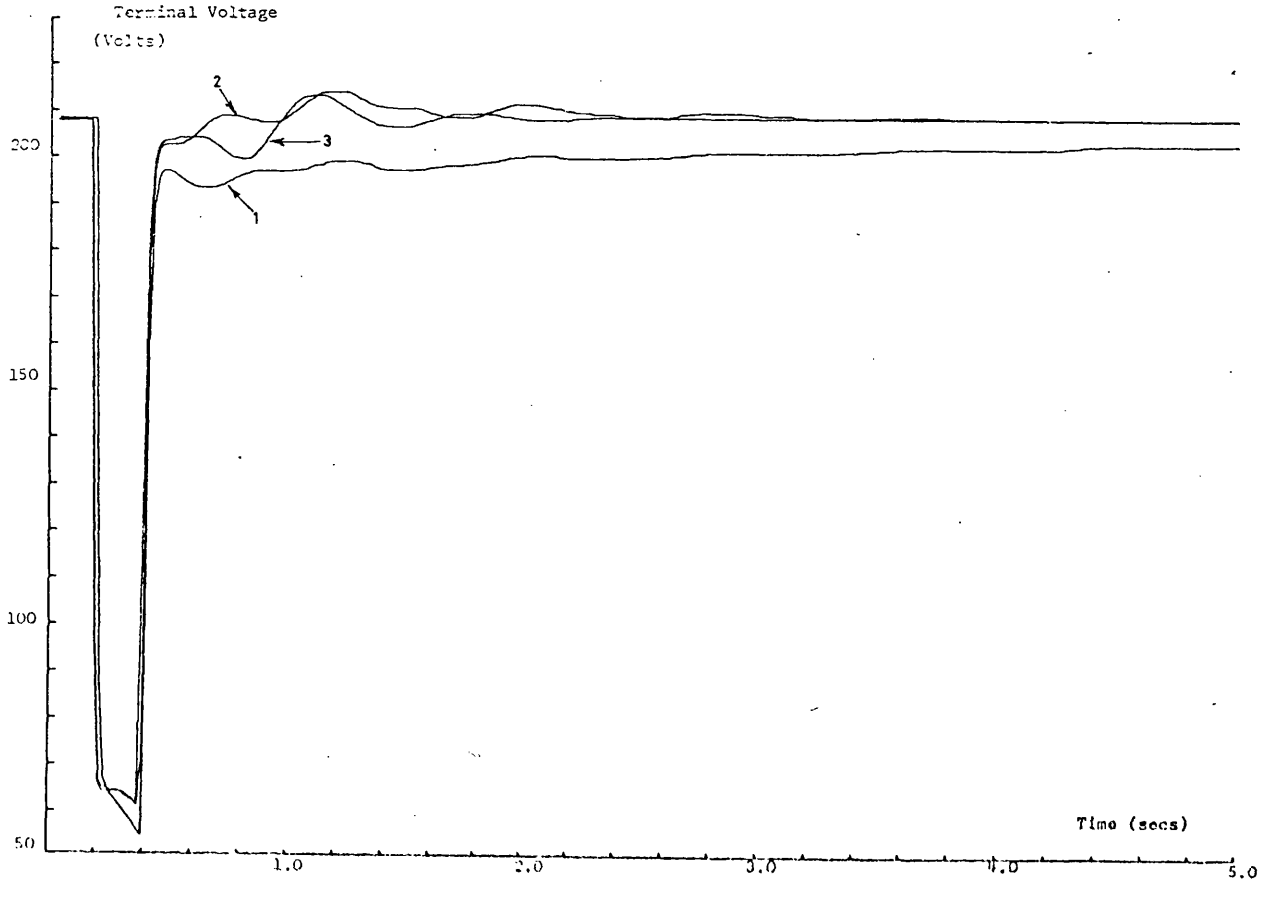


Fig. 8.12 Terminal Voltage against Time

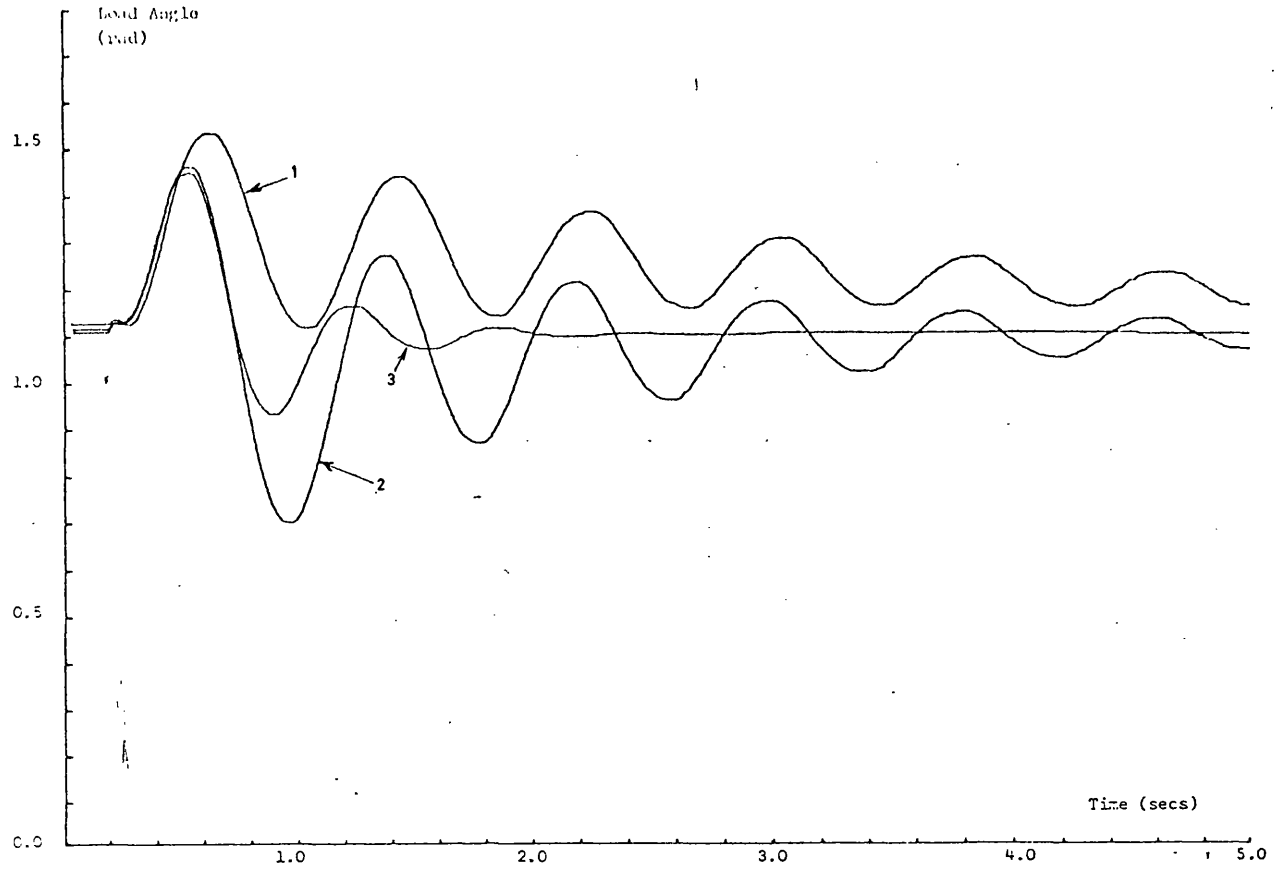


Fig. 8.13 Load Angle against Time

DETAILS OF FIGS. 8.11, 8.12 & 8.13

Configuration: Low gain (as Fig. 4.1)

Plot No. 1: Constant Excitation

Plot No. 2: AVR Control ($V_i = 0$)

Plot No. 3: AVR plus Digital State Feedback (V_i as eqn. 8.1)

Operating Point: B (Standard — Table 8.1)

Fault Duration: 220 mS

would apply a linear state feedback through a gain coefficient matrix which was dependent on the operating point of the system and the type of fault. It was considered that the validity of such a control system could be proven if three inter-dependent conditions were met. These conditions were that the computer would have sufficient storage area for the necessary feedback matrices, that there would be sufficient time to access these coefficients from the memory storage medium and that the resultant system performance would not be excessively sensitive to the feedback coefficients with respect to the quantisation of operating points. The first two conditions are largely dependent on the third in that a large quantisation of operating points means fewer matrices to store which also decreases the access time. It is considered that the following results demonstrate the viability of the system with the acceptance that further work in the field remains to be done.

8.3.1 Unity Power Factor Operation

8.3.1.1 Standard Fault Duration

The state feedback parameters were re-optimised for the system having the feedback into the input summing junction of the AVR as shown in Fig 4.2. As explained in that chapter, the sub-optimal state feedback law was found to be:

$$V_i = -0.564\ e'_q - 0.4613\ \delta + 0.0027\ p\delta \qquad (8.2)$$

when operating at point B (Table 8.1) for a standard fault. Tests were made under these conditions and the results are shown in Figs 8.14 to 8.22. These curves show the results of three types of control:

1. Constant Excitation

2. The AVR without any additional control signals

3. The AVR with the state feedback of eqn (8.2) applied at the high-gain point

Fig 8.15 shows that the recovery of terminal voltage from the faulted level was as rapid with the state feedback control as with the AVR control but was slightly less well regulated, as would be expected from the excitation control exercised. In fact, the slight overshoot of voltage on recovery may even be beneficial in allowing faster recovery of the station auxilaries. The rotor angle forward swing (in Fig 8.16) was limited to the same peak by both the AVR and state controllers due to them both applying maximum excitation on occurrence of the fault as shown in Fig 8.14. However, the subsequent oscillations of the rotor were uncontrolled by the AVR as it ceased to apply any significant control effort once the terminal voltage had recovered. These oscillations decayed at their natural frequency as shown by comparison to the rotor angle curve corresponding to the constant excitation control.

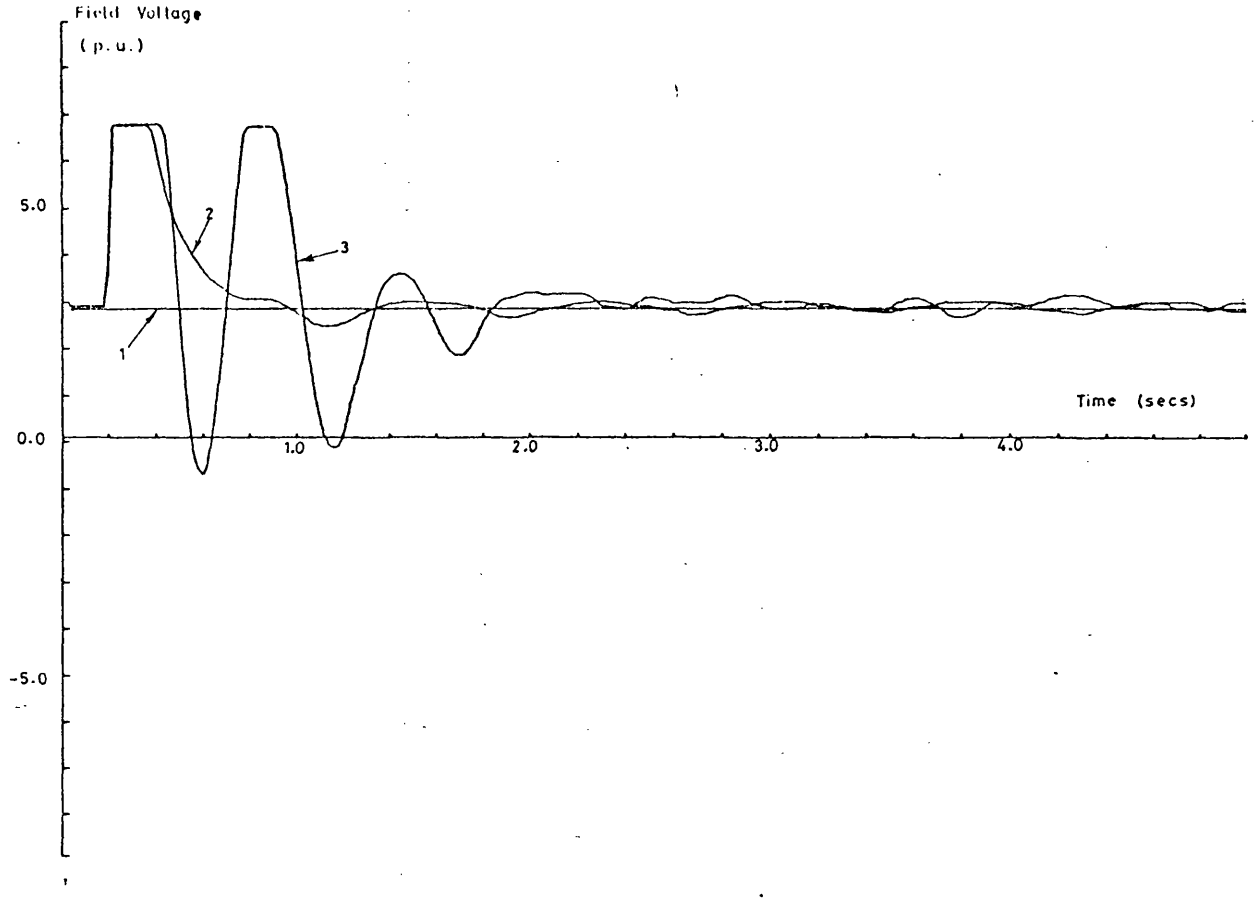


Fig. 8.14. Field Voltage (Excitation) against Time

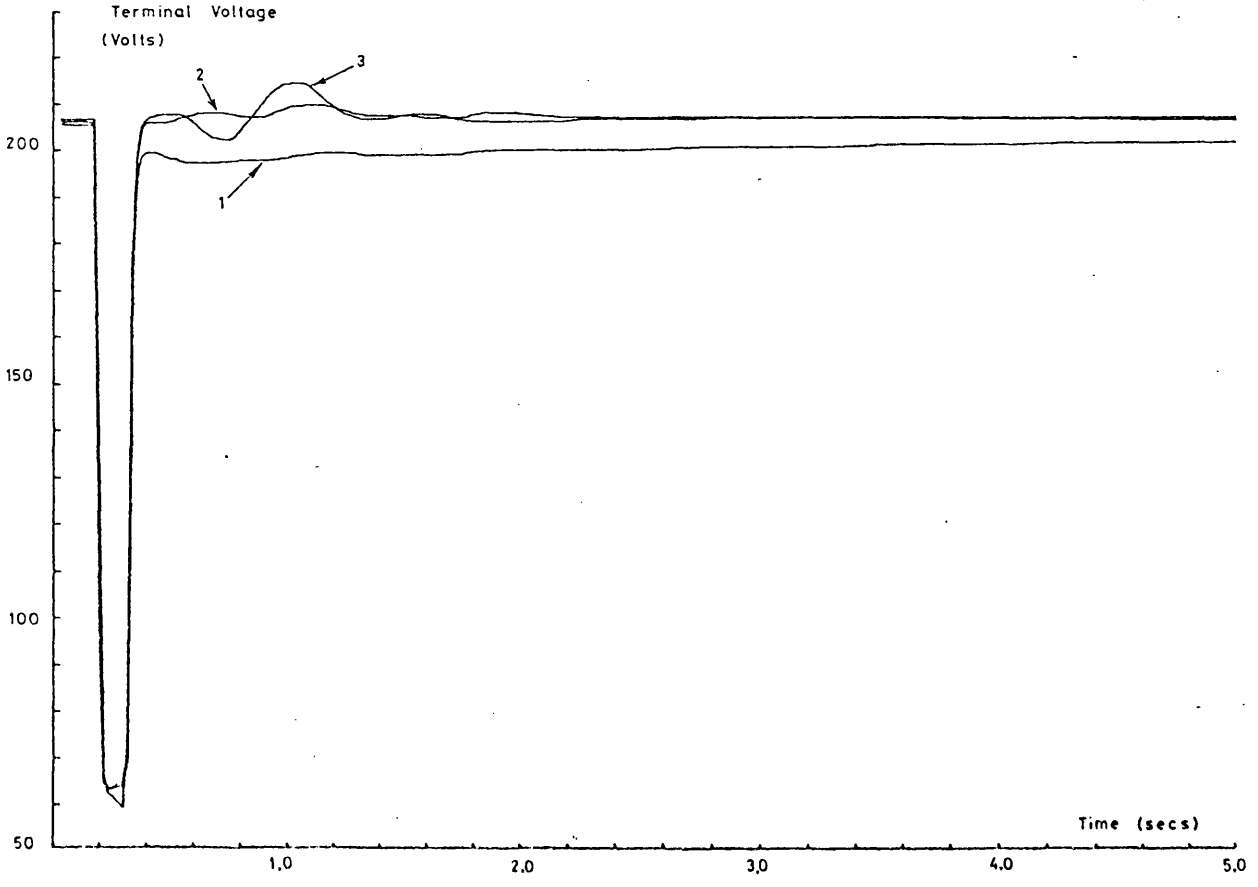


Fig. 8.15 Terminal Voltage against Time

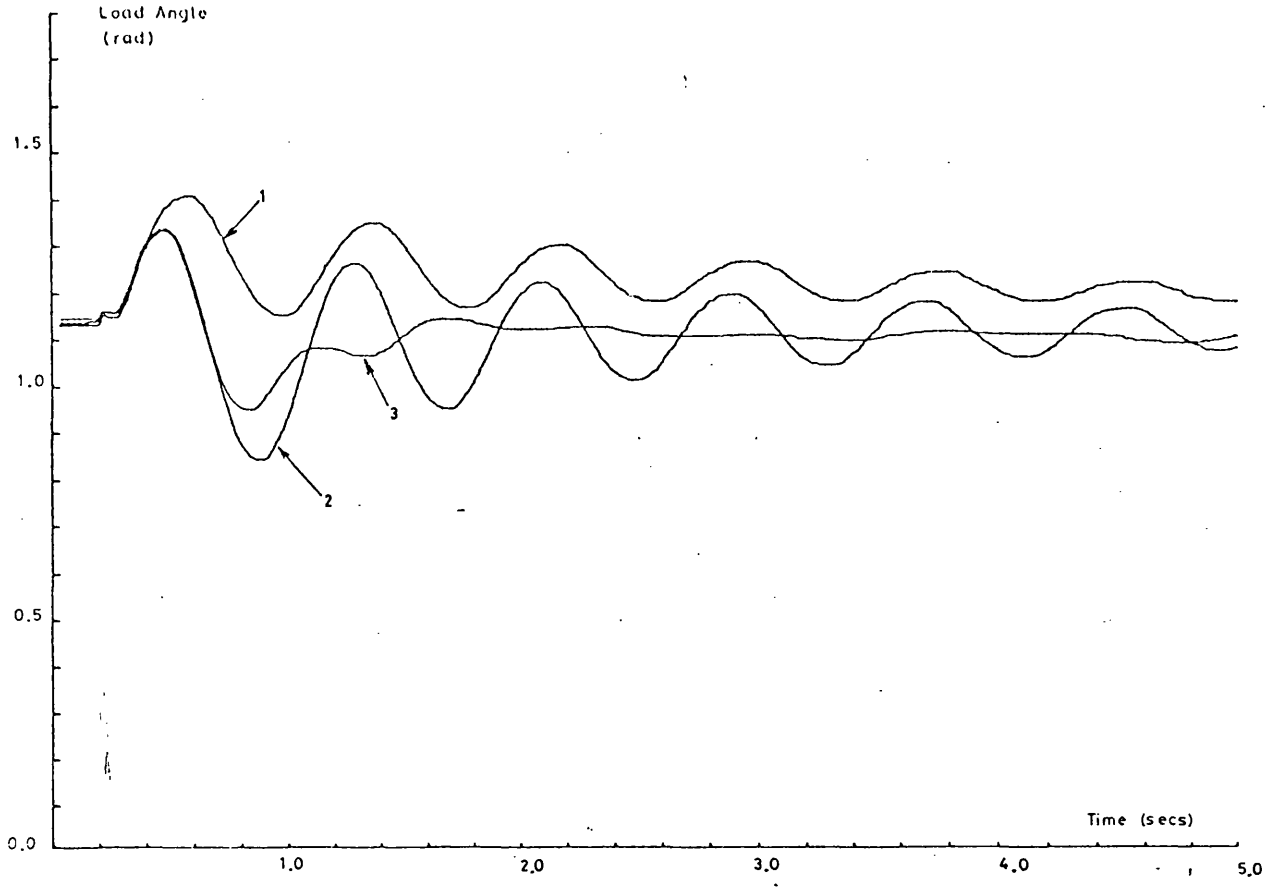


Fig. 8.16 Load Angle against Time

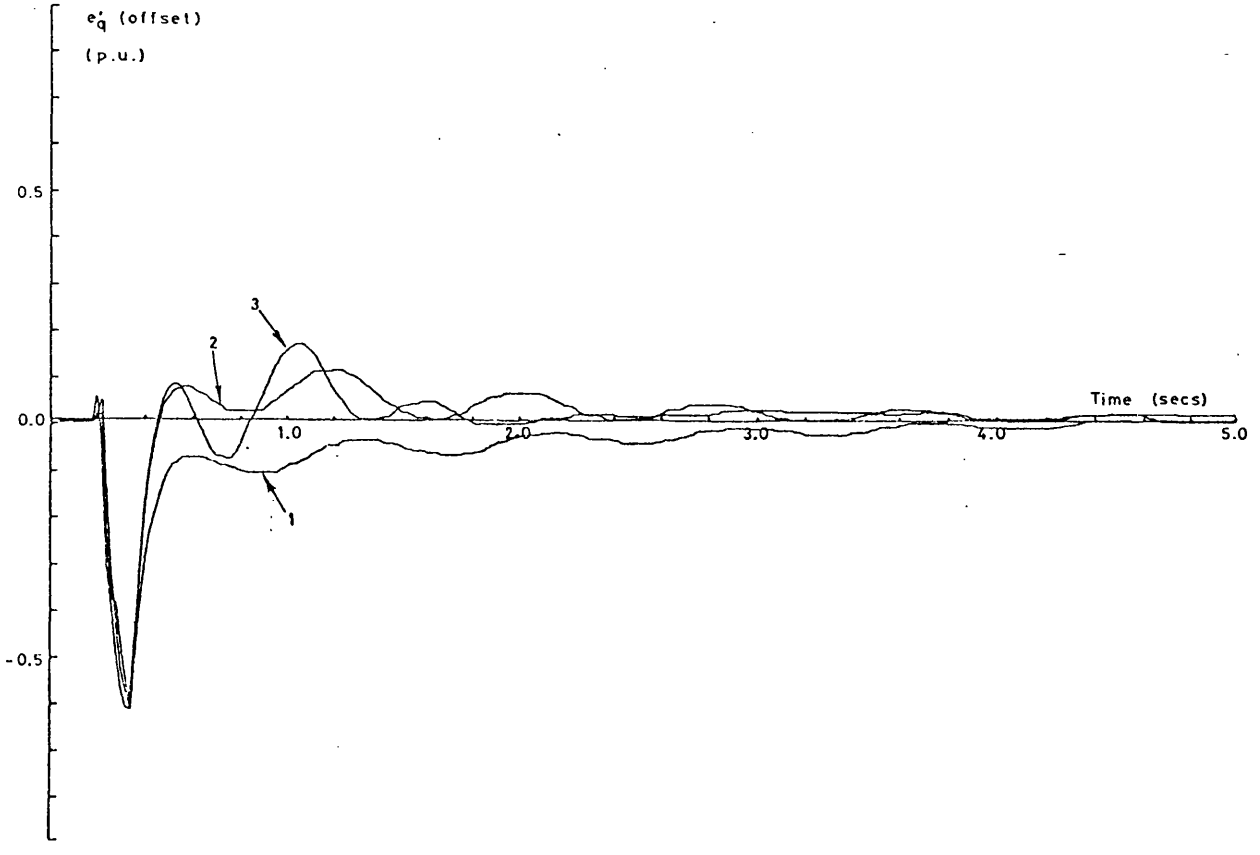


Fig. 8.17 E'_q (State Variable) against Time

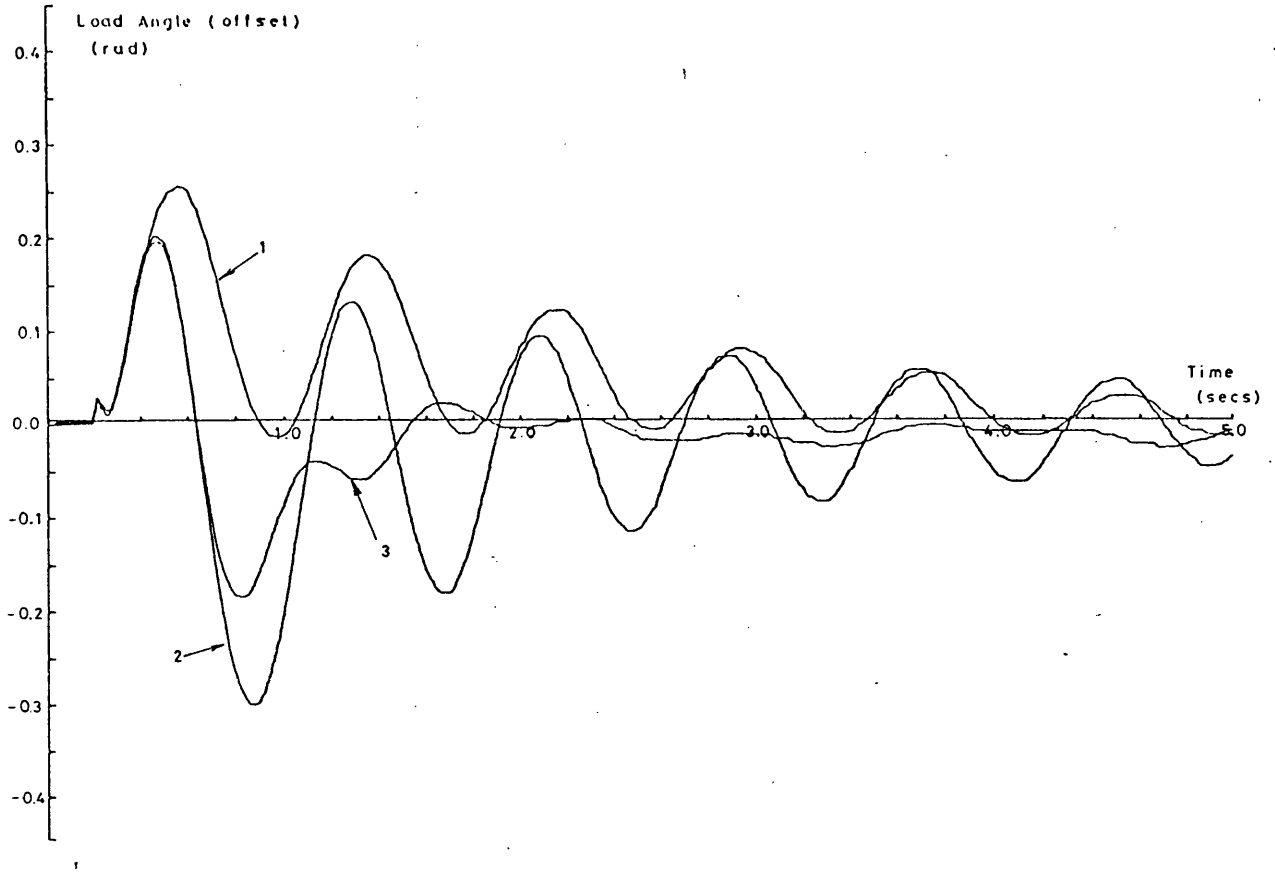


Fig. 8.18 Load Angle (State Variable) against Time

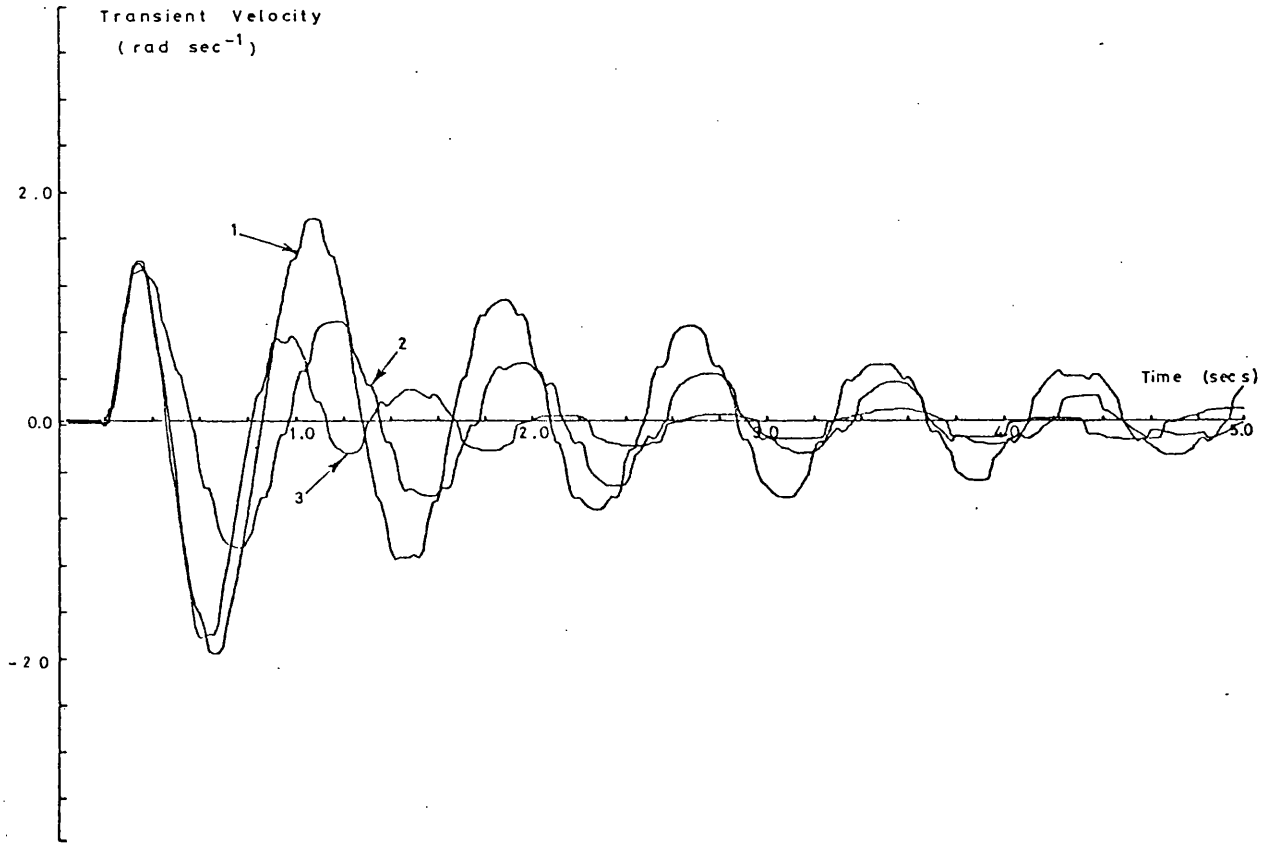


Fig. 8.19 Rotor Transient Velocity (State Variable) against Time

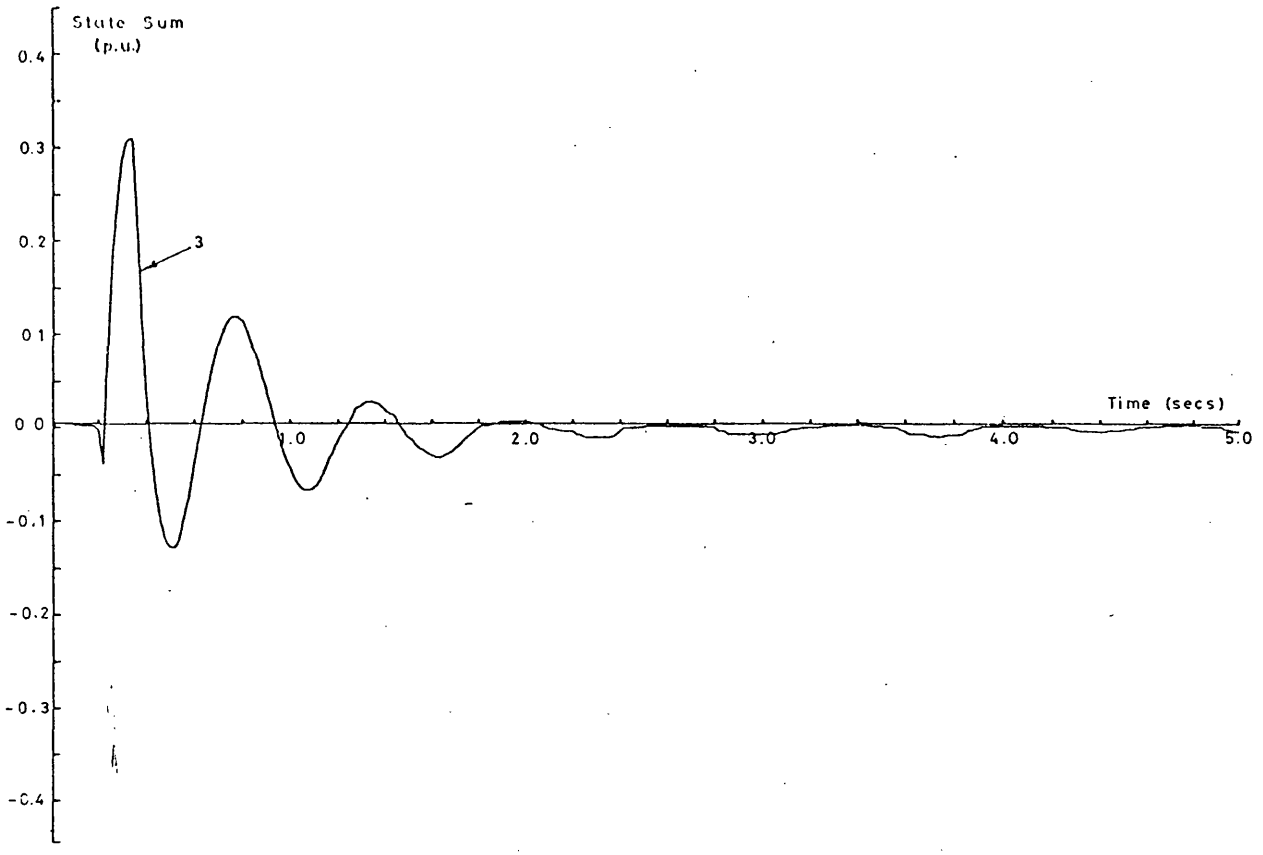


Fig. 8.20 State Feedback Signal (V_i) against Time

DETAILS OF FIGS. 8.14 TO 8.22

Configuration: High gain (as Fig. 4.2)

Plot No. 1: Constant Excitation

Plot No. 2: AVR Control ($V_i = 0$)

Plot No. 3: AVR plus Digital State Feedback (V_i as eqn. 8.2)

Operating Point: B (Standard — Table 8.1)

Fault Duration: 140 mS

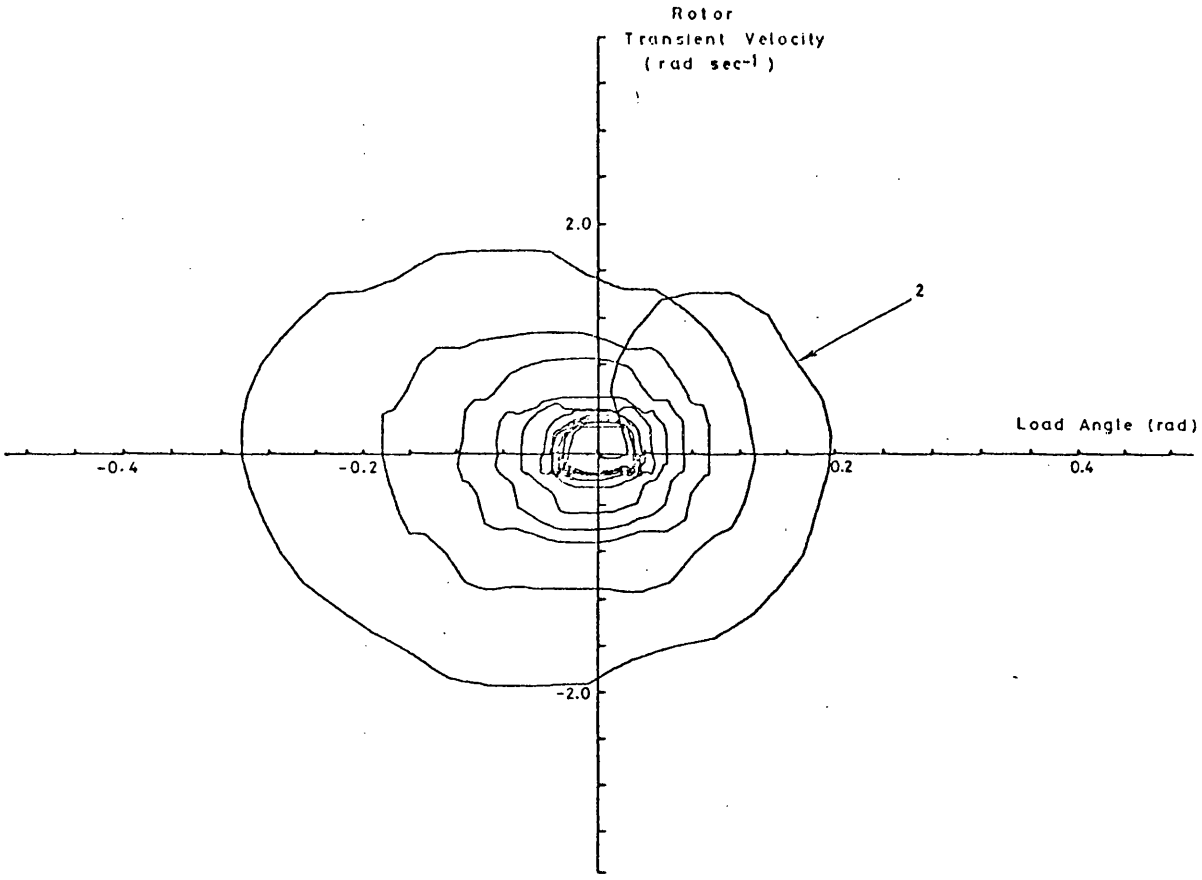


Fig. 8.21 Phase-Plane Plot of Load Angle against Transient Velocity

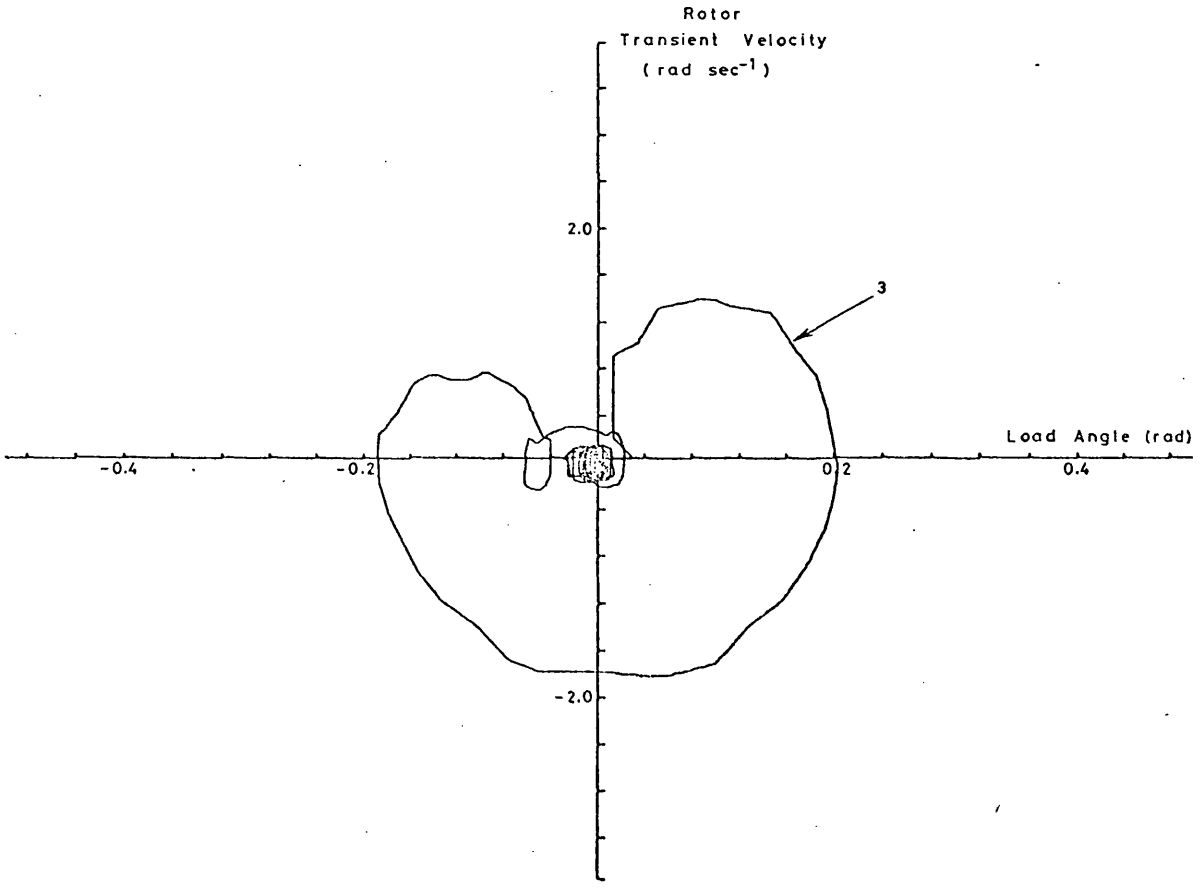


Fig. 8.22 Phase-Plane Plot of Load Angle against Transient Velocity

The characteristics of the three feedback states during the fault condition are shown in Figs 8.17, 8.18 and 8.19. An examination of the relative magnitudes of these signals with the coefficients of eqn (8.2) shows that, for approximately the first 0.2 seconds following the faults occurrence, the e'_q signal is predominant in the state feedback law but, after that time, the $p\delta$ signal contributes the largest component to the feedback function. As the first part of the state feedback signal is swamped by the AVR action, it is mainly the $p\delta$ signal which controls the rotor angle oscillations corresponding to a form of velocity feedback. The input signal to the AVR from the state feedback control is shown in Fig 8.20.

The effect of the state feedback controller in reducing the rotor angle swings is further shown by comparison of Figs 8.21 and 8.22. Fig 8.21 shows a phase-plane plot of rotor angle against rotor transient velocity following the fault under the control of the AVR. As shown, the response was highly oscillatory and slowly decayed to a condition of dynamic equilibrium. By contrast, the response under the control of digital state feedback, shown in Fig 8.22, demonstrated that the system was more rapidly brought back to the steady state operating condition, indicating a system that was inherently more stable.

8.3.1.2 Other Fault Durations

In order to qualitatively demonstrate the insensitivity of the system response to nonsub-optimal state feedback coeffidents,

the control law of eqn (8.2) was applied when the system experienced faults of other than 140 mS, the standard fault duration. When a fault duration of only 80 mS is experienced, the terminal voltage and rotor angle responses are shown in Figs 8.24 and 8.25 respectively. Under these conditions, the excitation control law is as shown in Fig 8.23. From consideration of these curves, it can be seen that the rotor swings are well controlled and the terminal voltage response is not impaired.

The effectiveness of the same control law was also investigated when the system experienced a fault of 220 mS, this being the longest duration likely to be experienced in practice. The results are shown in Figs 8.26 to 8.28 and it can be seen that the digital state feedback controller has produced a significant improvement in the response by controlling the rotor angle swings yet not restricting the terminal voltage recovery. The terminal voltage recovers to its former level at point 'P' at approximately the same instant under the influence of both types of control, yet the rotor angle swings are significantly reduced by the use of the state feedback controller.

8.3.2 Lagging Volt-Ampere Reactive Region

Further tests of the effectiveness of the control law by eqn (8.2) were also made with the synchronous generator in an over-excited condition, i.e. supplying lagging VARs to the system. The results of tests using an 80 mS fault duration of

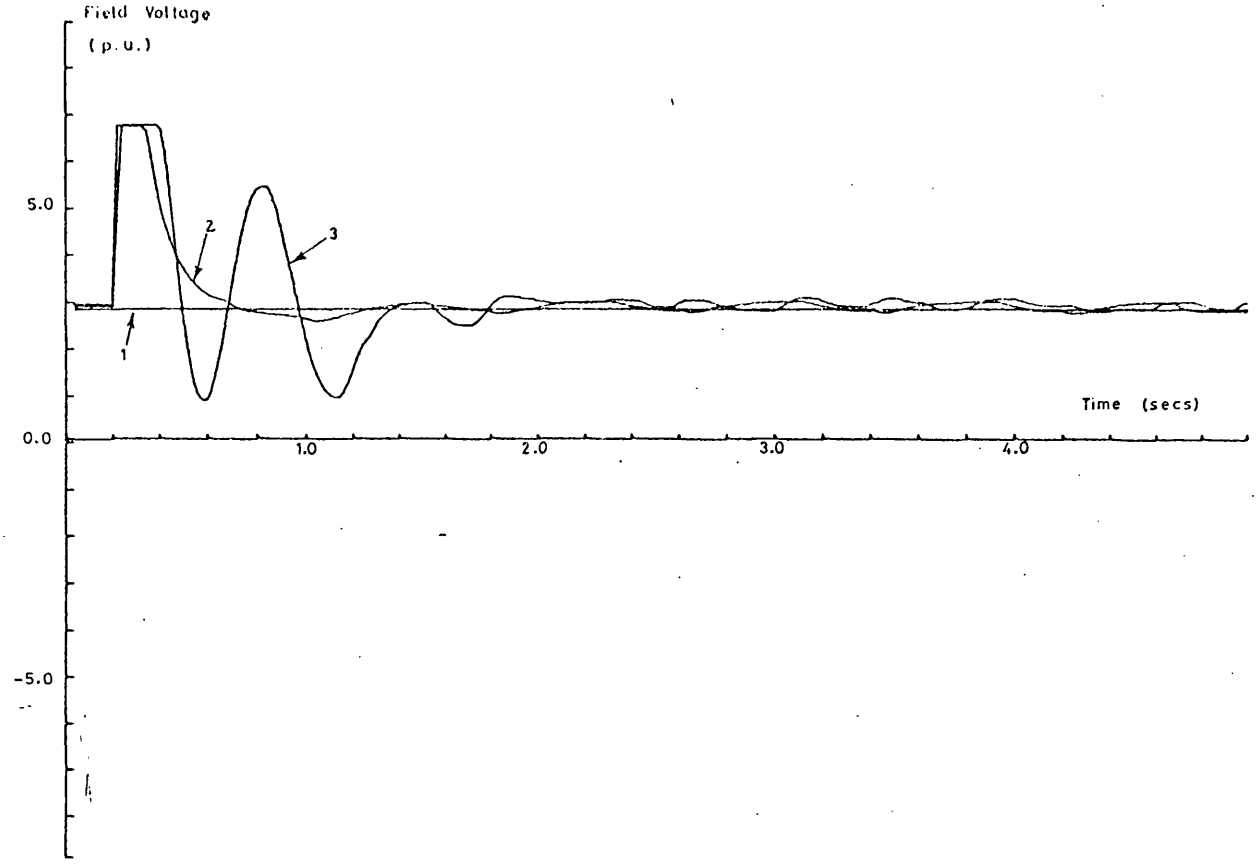


Fig. 8.23 Field Voltage (Excitation) against Time

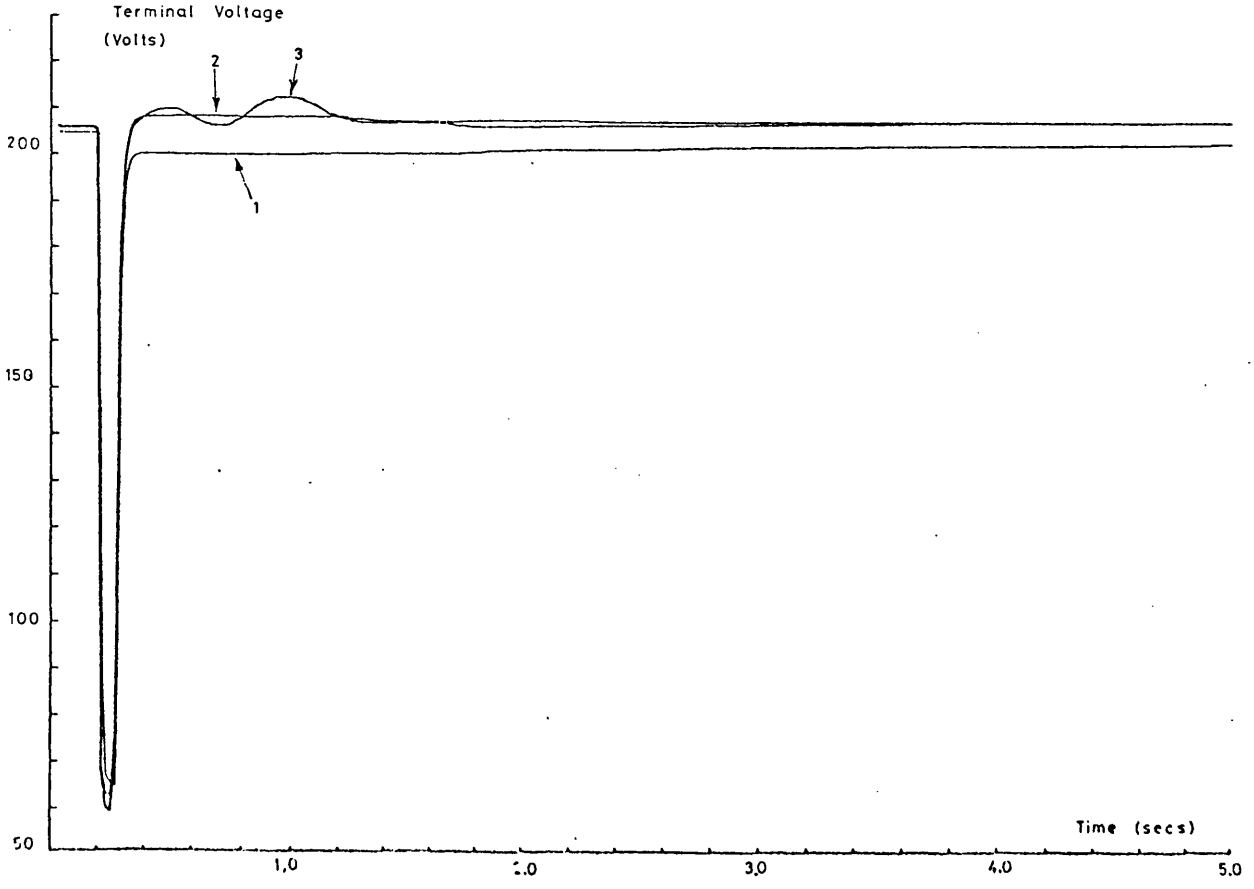


Fig. 8.24 Terminal Voltage against Time

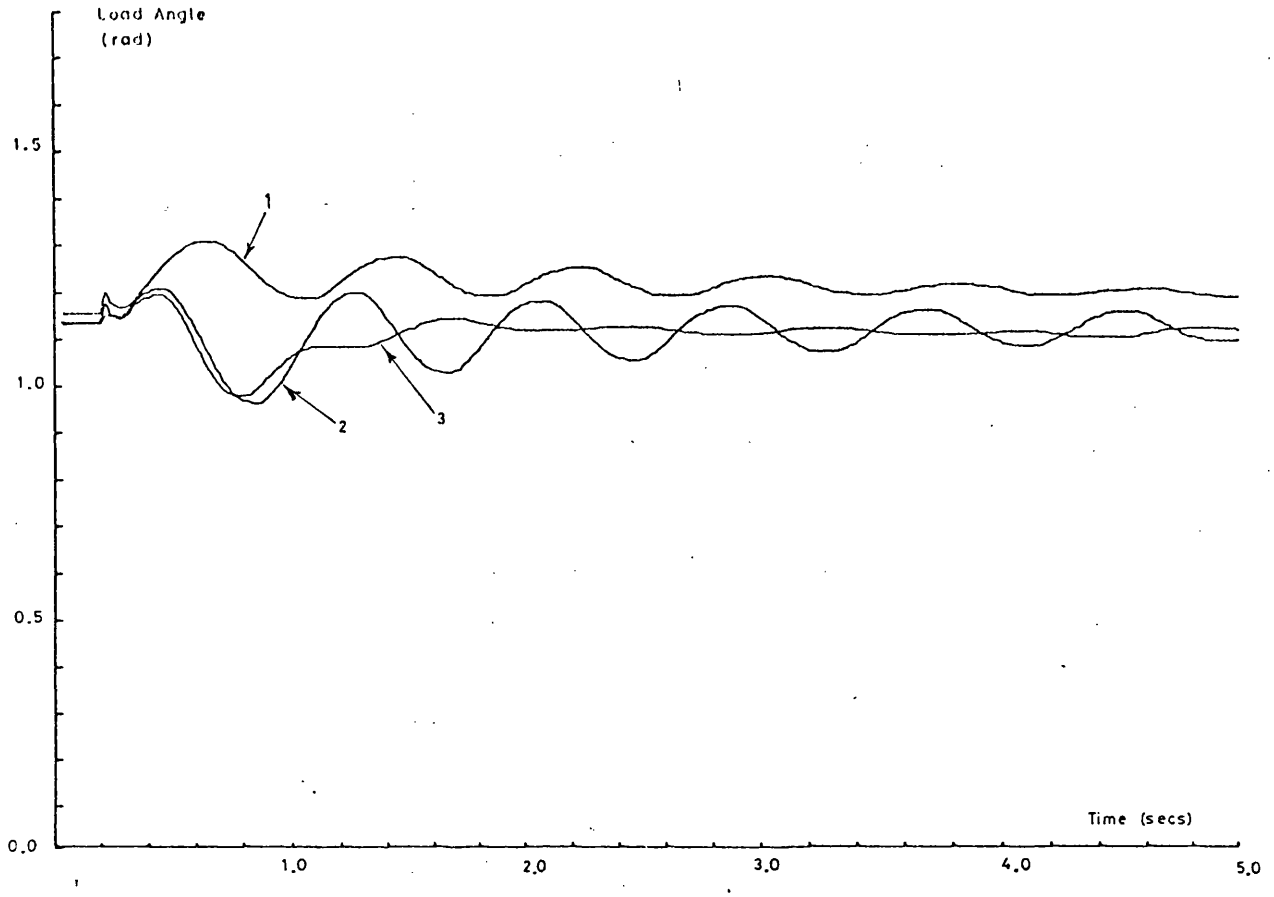


Fig. 8.25 Load Angle against Time

DETAILS OF FIGS. 8.23 , 8.24 & 8.25

Configuration: High gain (as Fig. 4.2)

Plot No. 1: Constant Excitation

Plot No. 2: AVR Control ($V_i = 0$)

Plot No. 3: AVR plus Digital State Feedback (V_i as eqn. 8.2)

Operating Point: B (Standard — Table 8.1)

Fault Duration: 80 mS

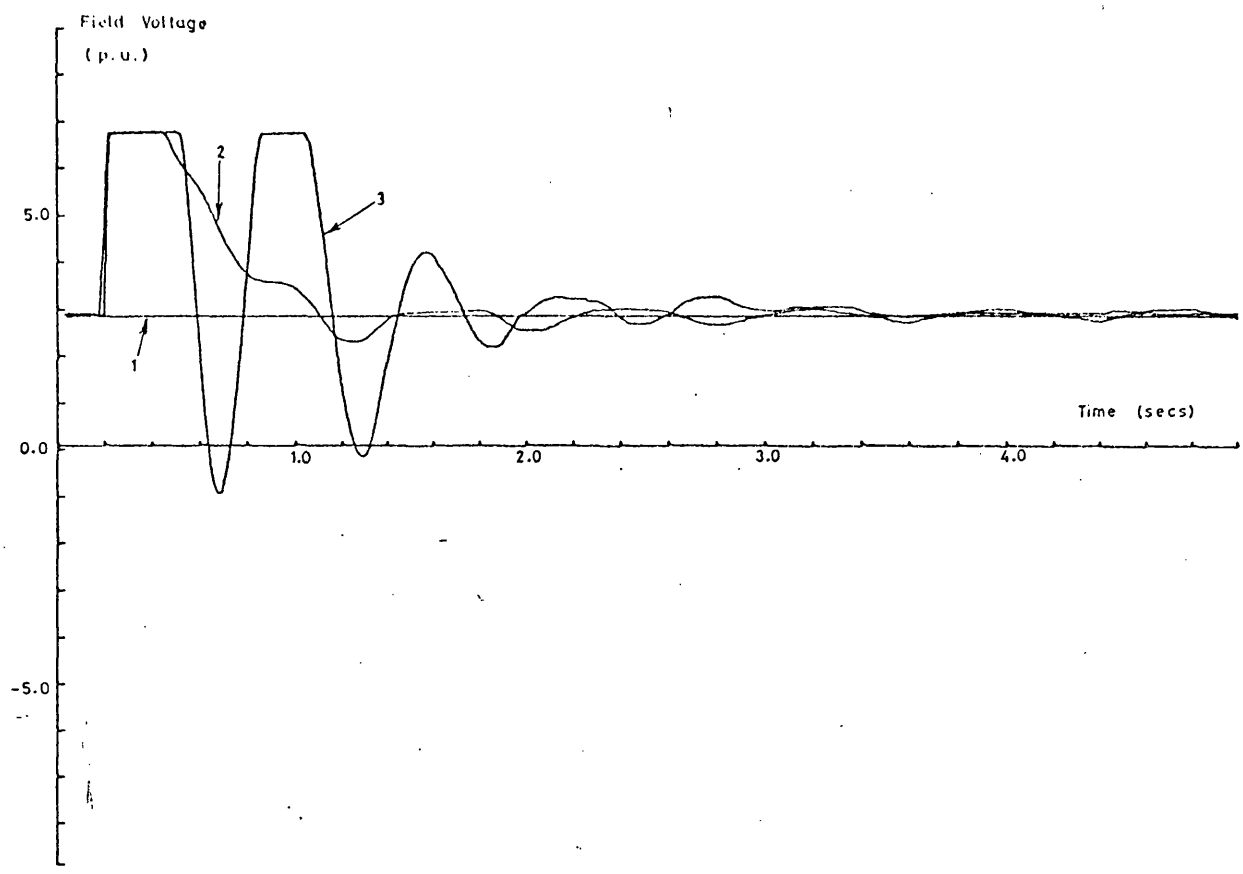


Fig. 8.26 Field Voltage (Excitation) against Time

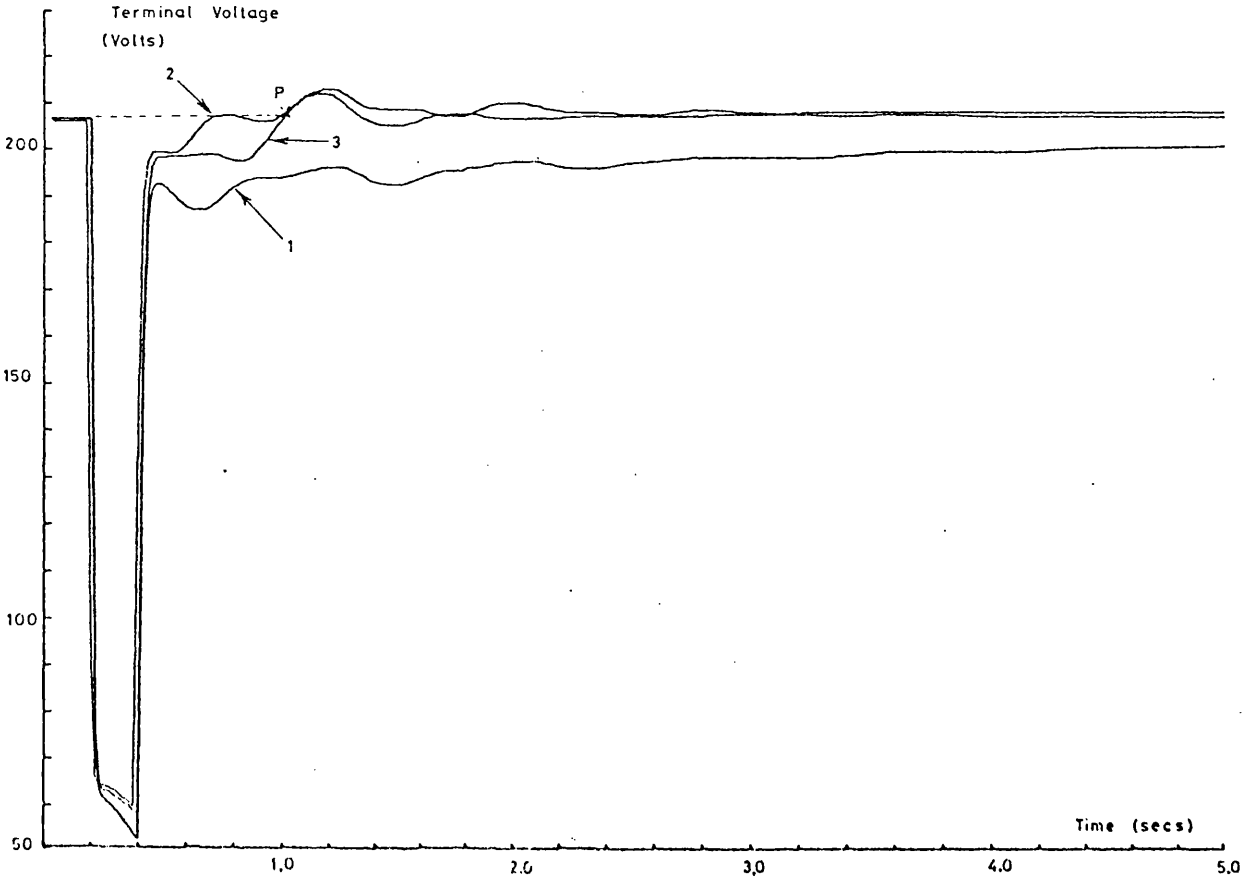


Fig. 8.27 Terminal Voltage against Time

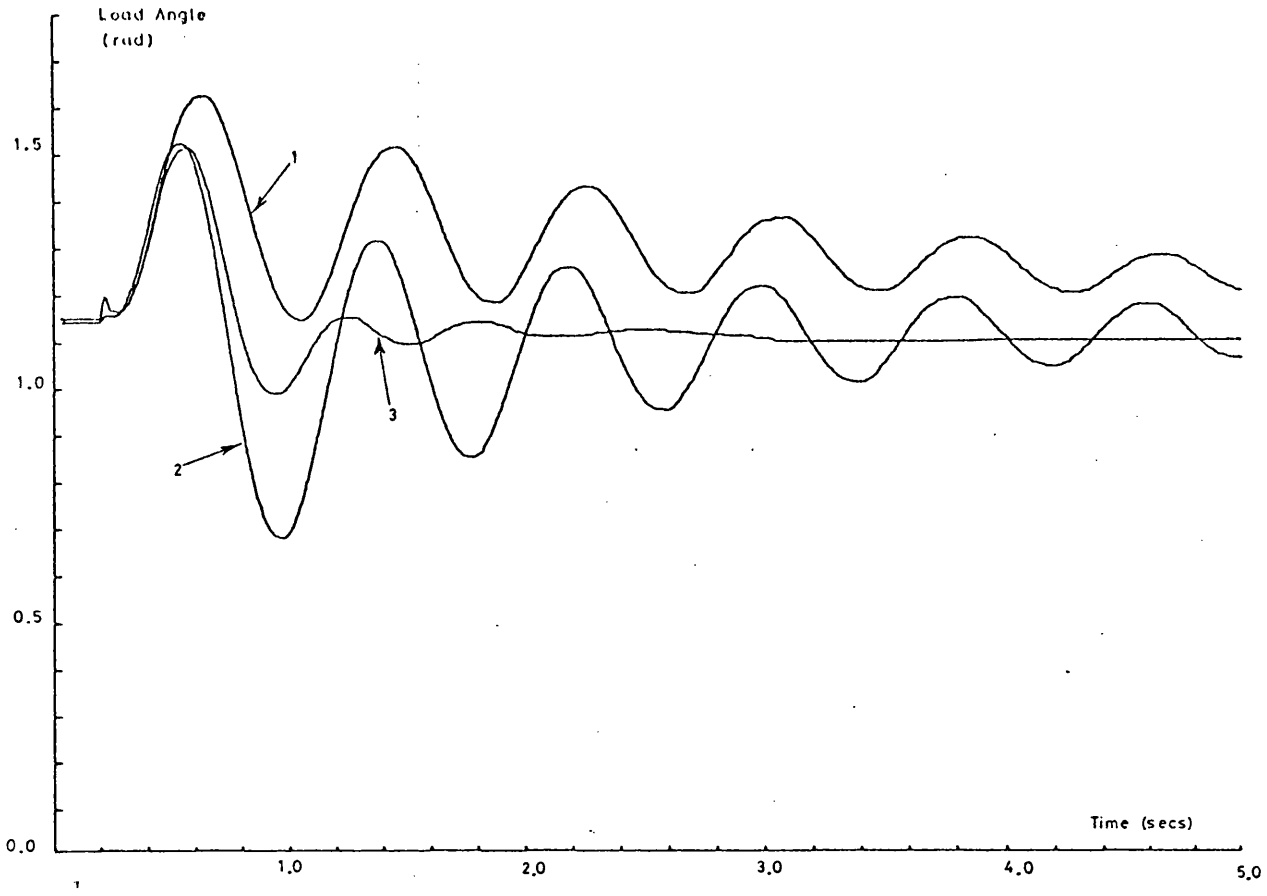


Fig. 8.28 Load Angle against Time

DETAILS OF FIGS. 8.26, 8.27 & 8.28

Configuration: High gain (as Fig. 4.2)

Plot No. 1: Constant Excitation

Plot No. 2: AVR Control ($V_i = 0$)

Plot No. 3: AVR plus Digital State Feedback (V_i as eqn. 8.2)

Operating Point: B (Standard — Table 8.1)

Fault Duration: 220 mS

the type previously described with the system operating at point 'A' (Table 8.1) are shown in Figs 8.29 to 8.31. Again, a significant improvement in rotor angle response was shown by the inclusion of the state feedback control of eqn (8.2) even though this has been optimised for a different operating condition.

When the fault duration was extended to 140 mS, the system response was as shown in Figs 8.32 to 8.34. The load angle swings, shown in Fig 8.34, were reduced to zero virtually after one forward and one reverse swing, a considerable improvement on the AVR-controlled response, which was highly oscillatory. Again, the first peak of the response in both cases was almost identical in that both controllers had very similar characteristics initially, as shown in Fig 8.32.

A fault of 220 mS is the most onerous that is likely to be experienced in practice and the response of the system to a fault of this duration when operating in a lagging VAR region is shown by Figs 8.35 to 8.37. Fig 8.36 shows that the terminal voltage recovery to point 'P' is as rapid for both types of control shown. However, Fig 8.37 shows a very marked improvement in the rotor angle response, there being only a very slight backswing in the rotor angle following the forward swing when controlled by the state feedback control system.

8.3.3 Leading Volt-Ampere Reactive Region

The synchronous generator running in an under-excited condition

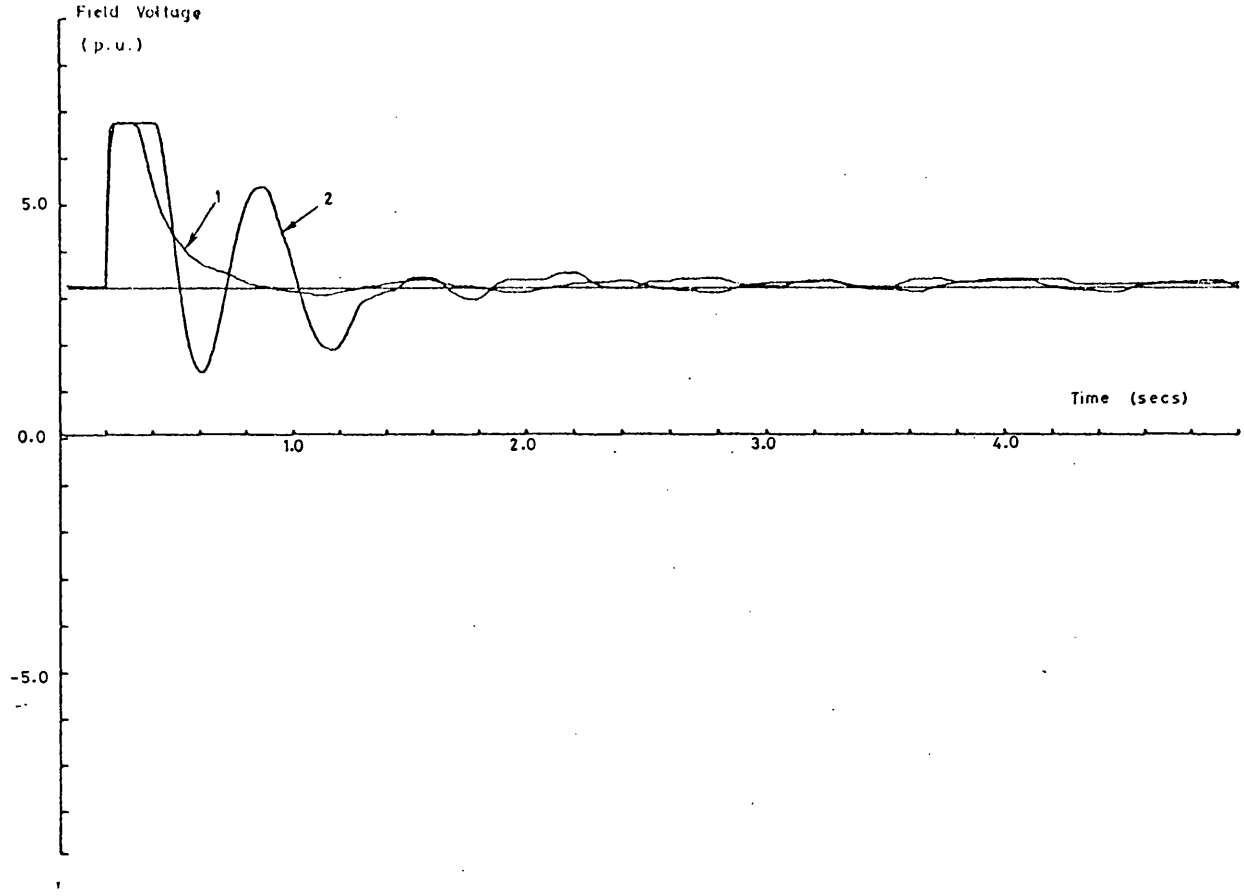


Fig. 8.29 Field Voltage (Excitation) against Time

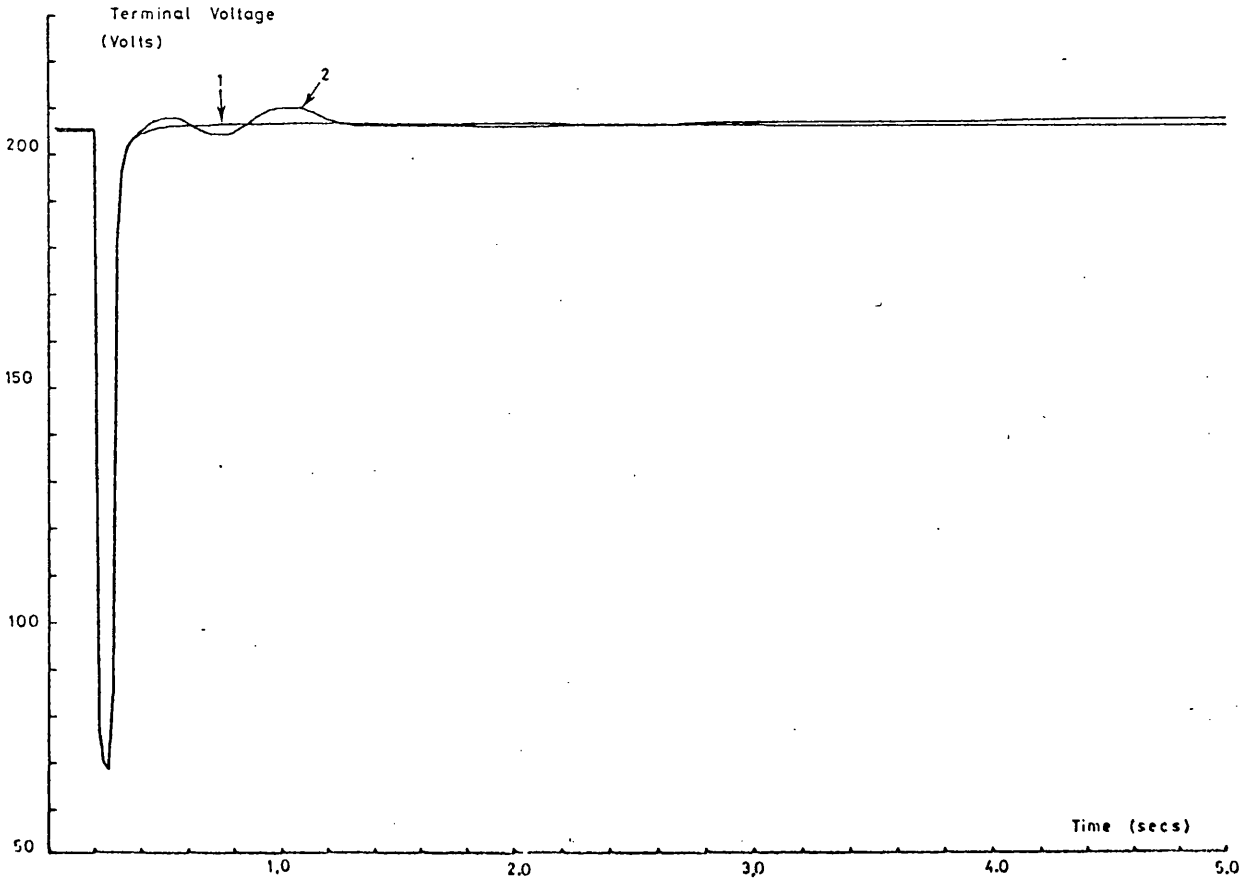


Fig. 8.30 Terminal Voltage against Time

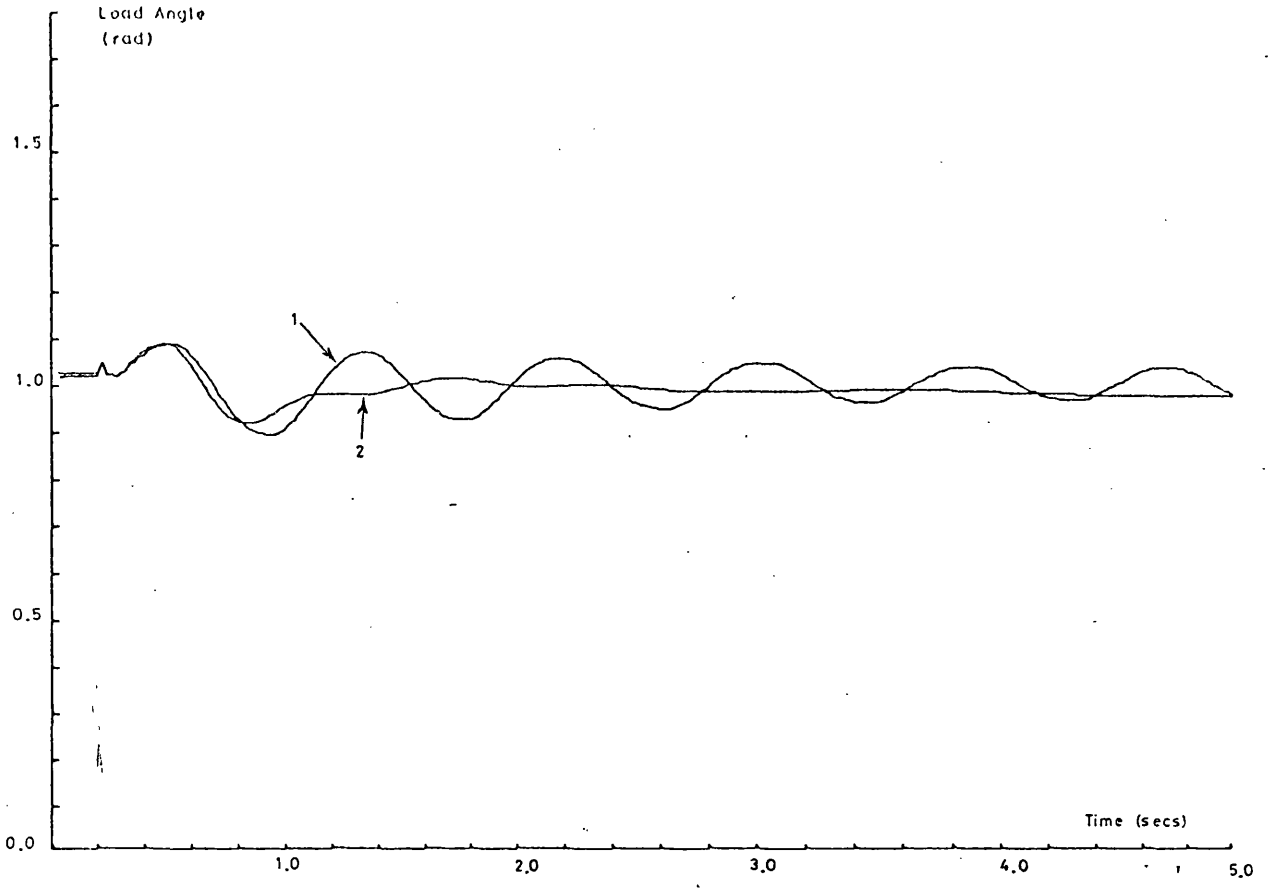


Fig. 8.31 Load Angle against Time

DETAILS OF FIGS. 8.29 , 8.30 & 8.31

Configuration : High gain (as Fig. 4.2)

Plot No. 1 : AVR Control ($V_i = 0$)

Plot No. 2 : AVR plus Digital State Feedback (V_i as eqn. 8.2)

Operating Point : A (Lagging VARs —— Table 8.1)

Fault Duration : 80 mS

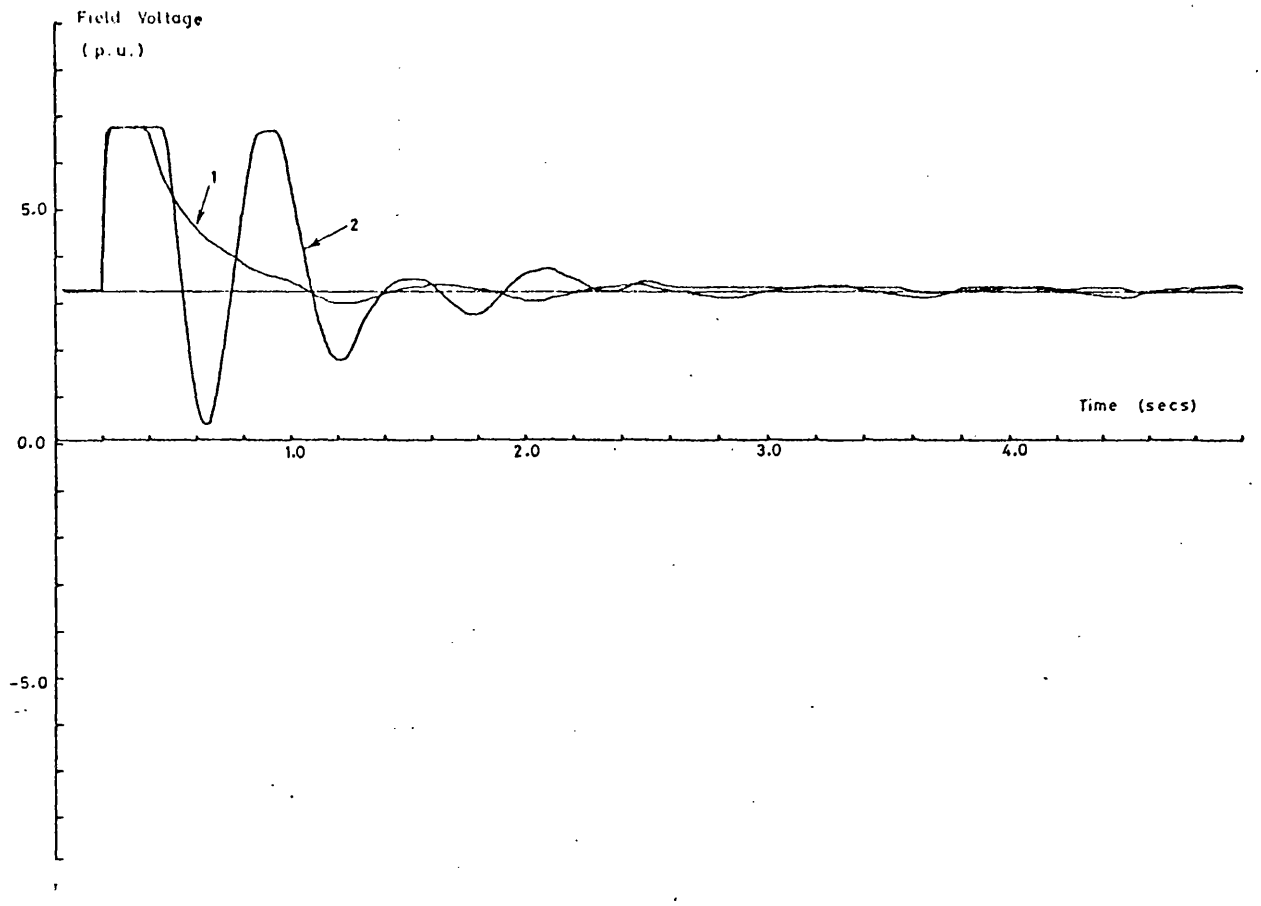


Fig. 8.32 Field Voltage (Excitation) against Time

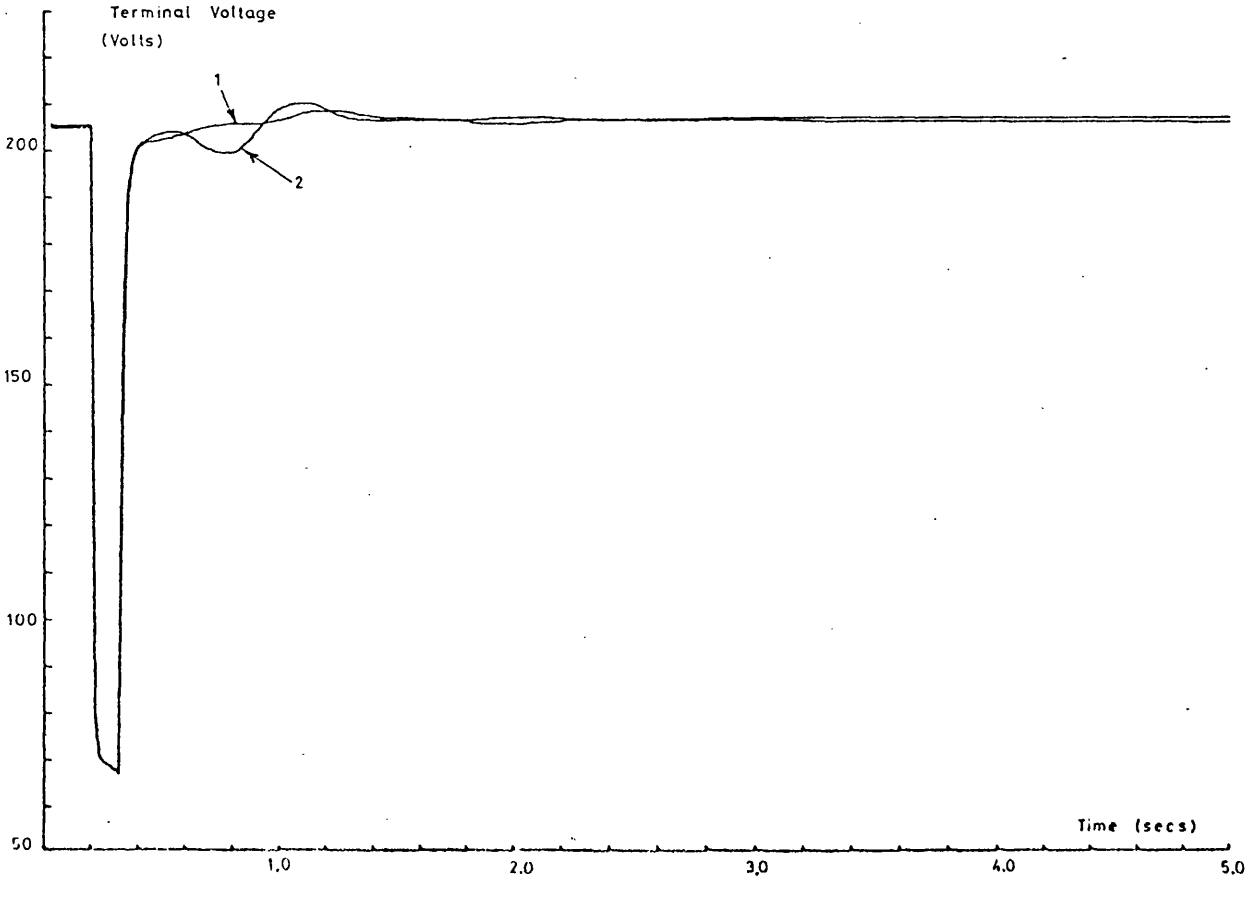


Fig. 8.33 Terminal Voltage against Time

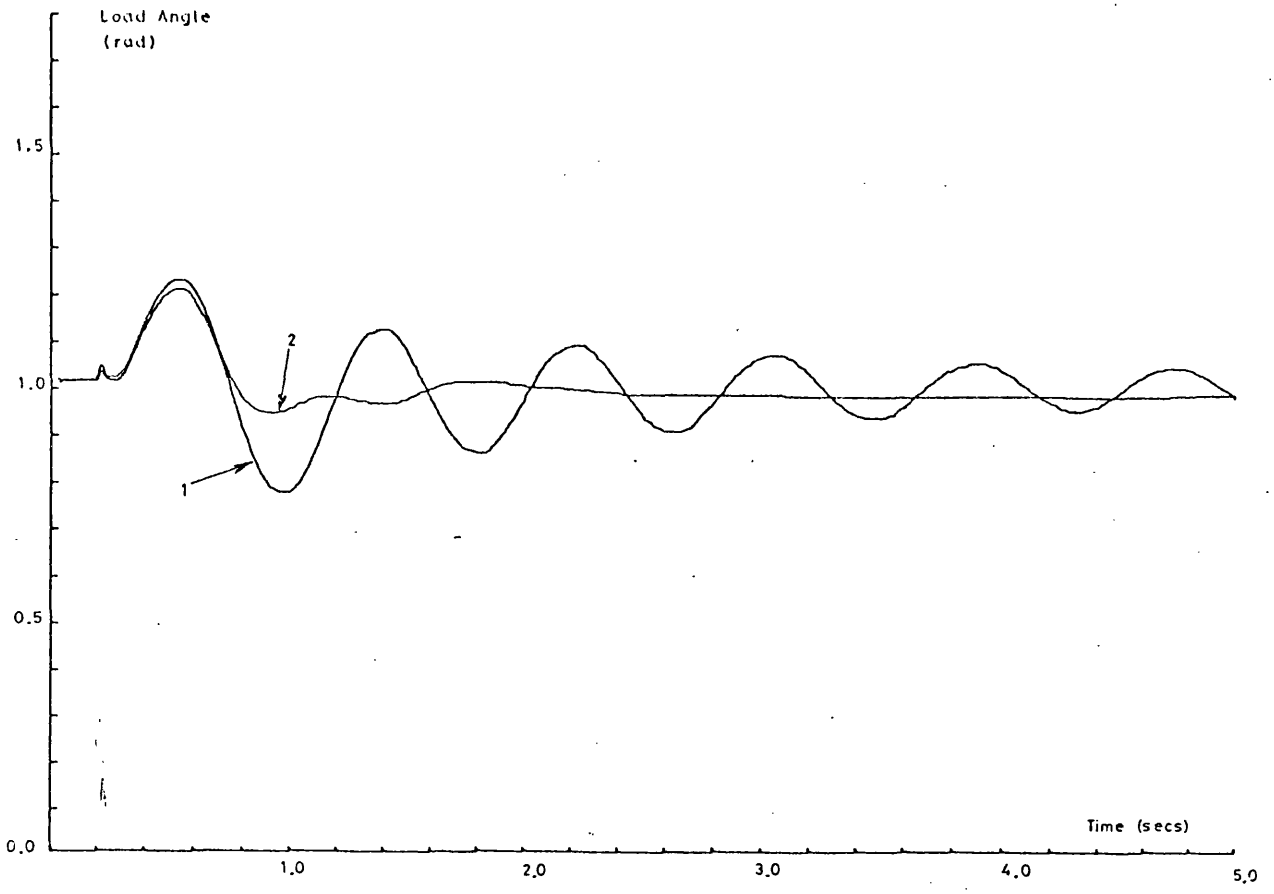


Fig. 8.34 Load Angle against Time

DETAILS OF FIGS. 8.32 , 8.33 & 8.34

Configuration : High gain (as Fig. 4.2)

Plot No. 1 : AVR Control ($V_i = 0$)

Plot No. 2 : AVR plus Digital State Feedback (V_i as eqn. 8.2)

Operating Point : A (Lagging VARs — Table 8.1)

Fault Duration : 140 mS

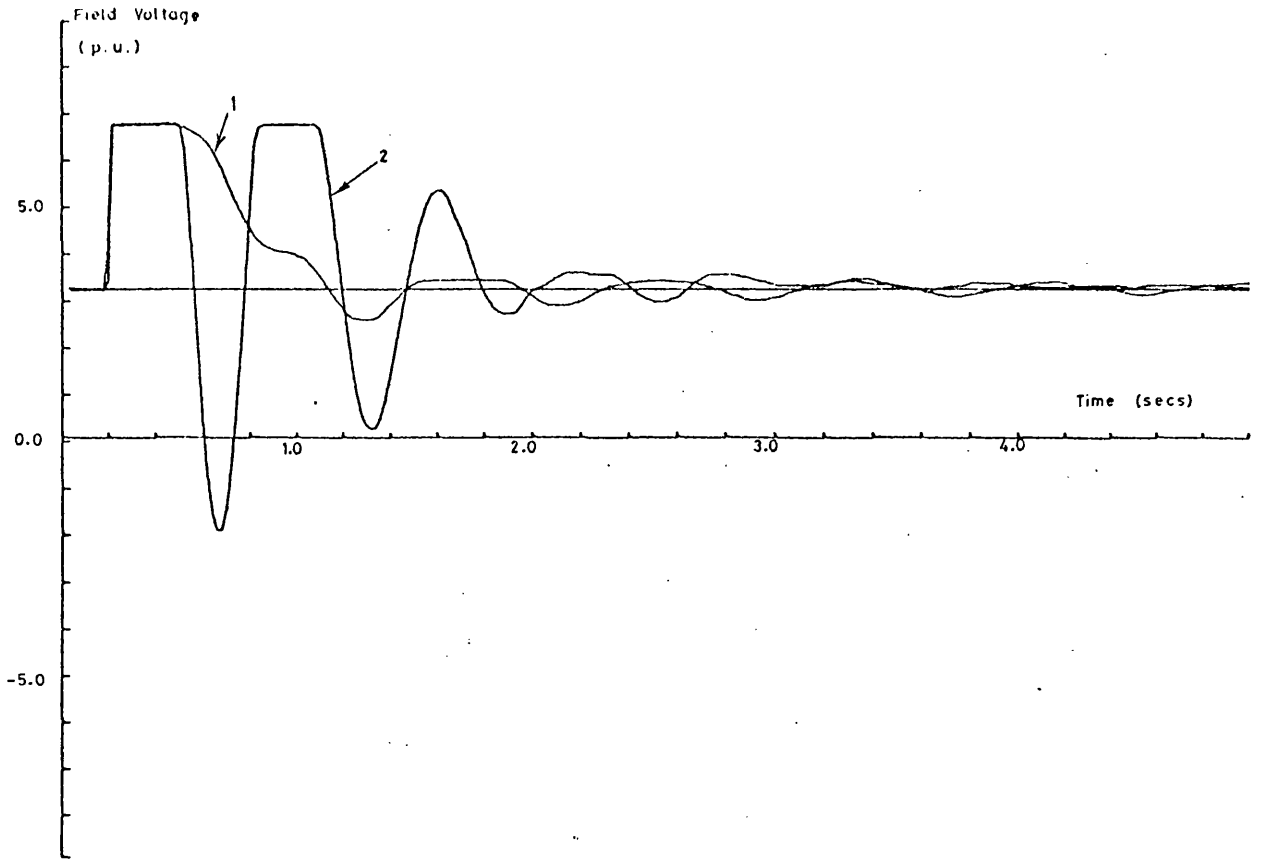


Fig. 8.35 Field Voltage (Excitation) against Time

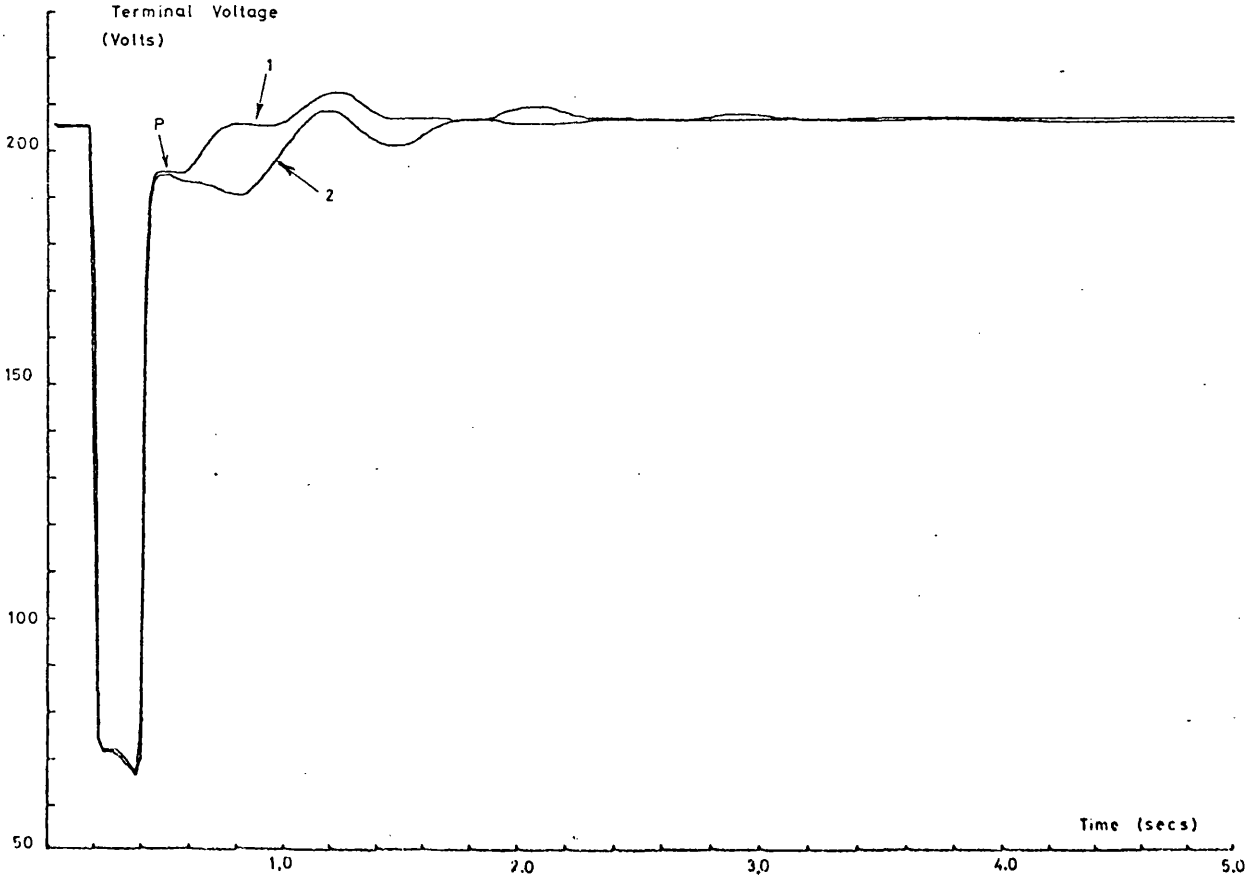


Fig. 8.36 Terminal Voltage against Time

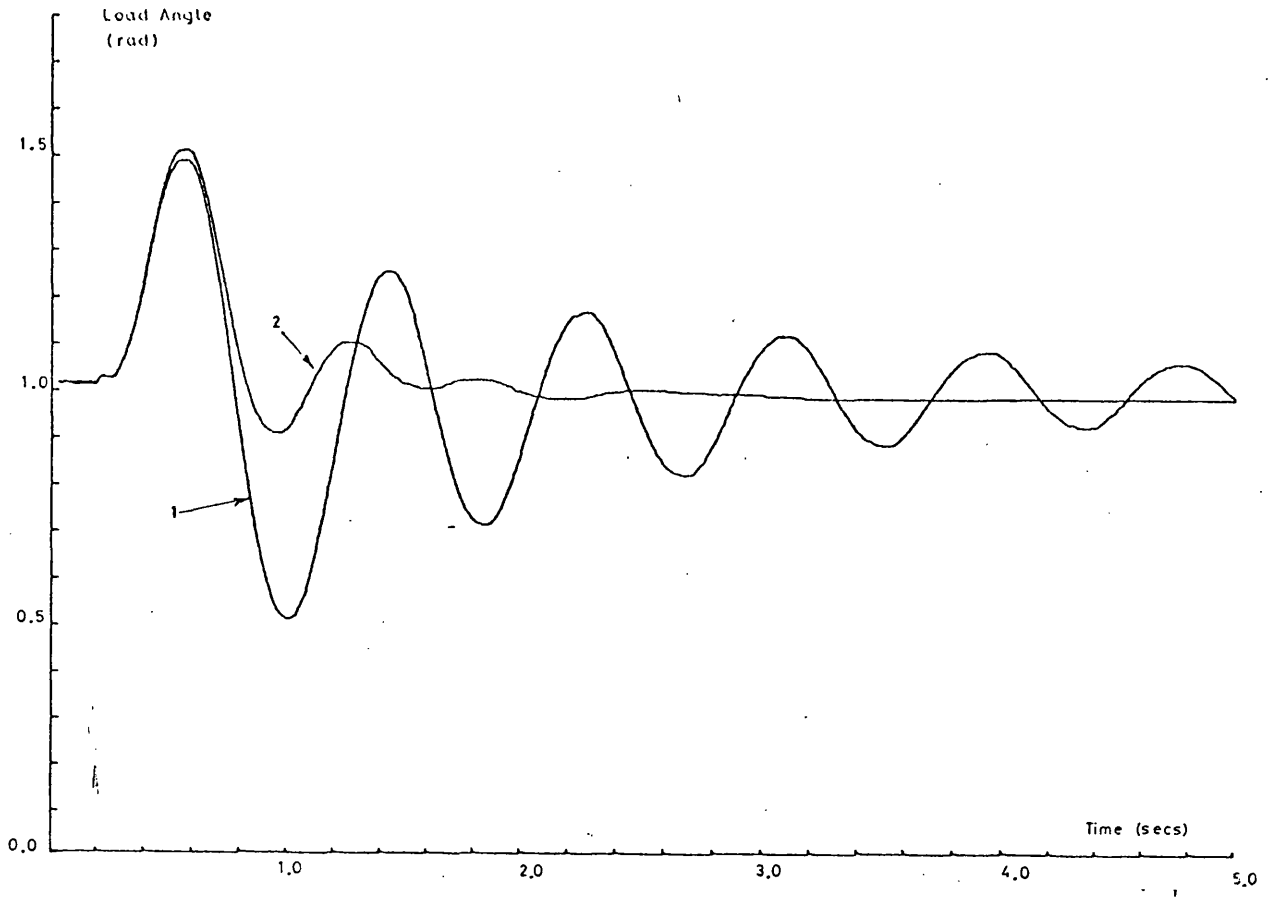


Fig. 8.37 Load Angle against Time

DETAILS OF FIGS. 8.35 , 8.36 & 8.37

Configuration: High gain (as Fig. 4.2)

Plot No. 1: AVR Control ($V_i = 0$)

Plot No. 2: AVR plus Digital State Feedback (V_i as eqn. 8.2)

Operating Point: A (Lagging VARs — Table 8.1)

Fault Duration: 220 mS

and supplying leading VARs to the system is running in a condition nearest to its dynamic stability limit. Thus, the effect of the state variable feedback control system in reducing the rotor angle swings in this region is of great interest.

Figs 8.38 to 8.40 show the response of the system when an 80 mS fault is experienced when operating in Region C (Table 8.1). However, as the 80 mS fault is not the most onerous which can occur, comment will be reserved until later. Figs 8.41 to 8.43 show the response under the same conditions for a fault of 140 mS duration and Figs 8.44 to 8.46 show the behaviour of the same system variables under the influence of a 220 mS fault.

Examination of these results shows that the turbo-alternator system is normally operating at a load angle of 1.32 rad (76^{o}) under these conditions and that as the fault duration is increased from 80 to 140 to 220 mS the peak of the first swing advances from 1.38 rad (79^{o}) to 1.50 rad (86^{o}) and 1.70 rad (97^{o}) respectively. Even with a 220 mS fault duration the system is still stable and the rotor angle oscillations are quickly reduced as shown in Fig 8.46. Under these worst-case conditions, the terminal voltage is less well regulated and experiences a slightly greater overshoot as shown in Fig 8.45. Under shorter fault durations the deviation from the AVR-controlled response is significantly less yet the improvement in rotor angle performance is marked.

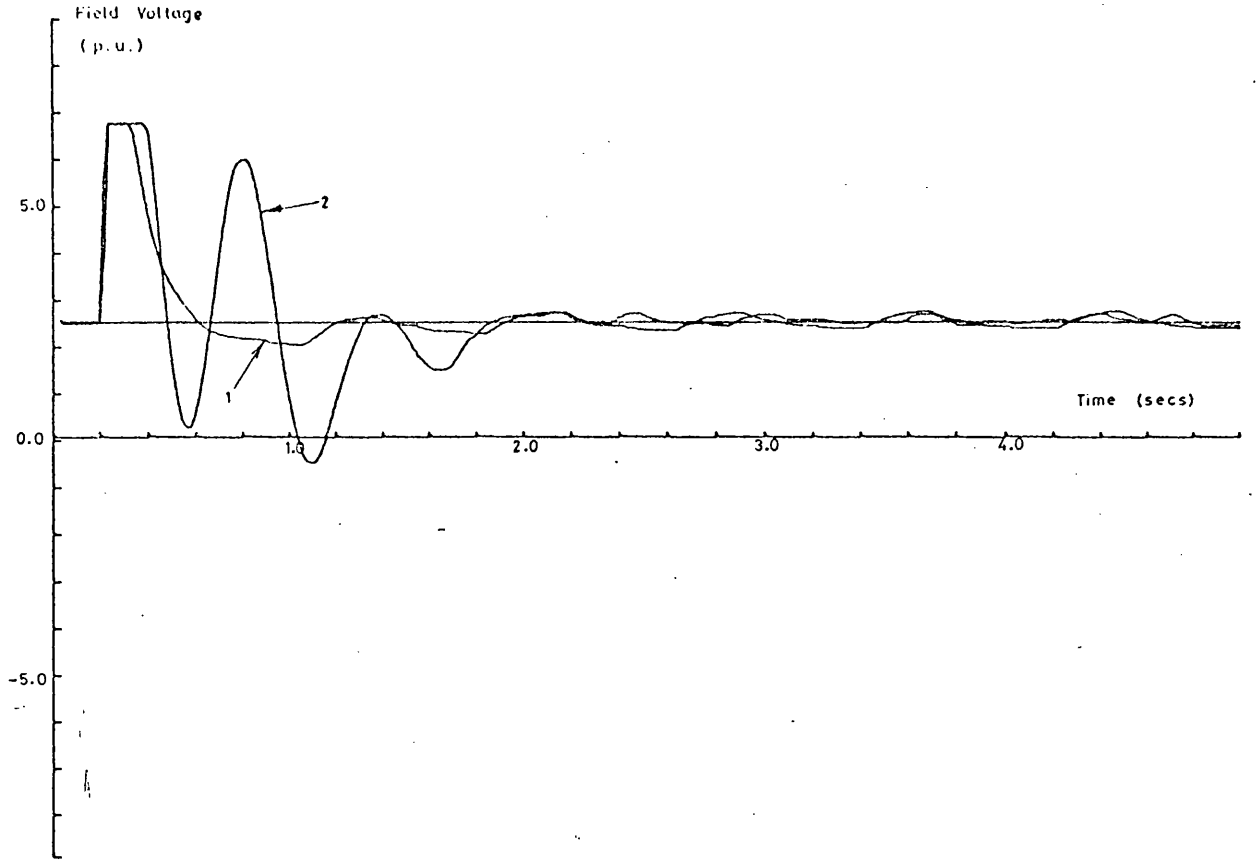


Fig. 8.38 Field Voltage (Excitation) against Time

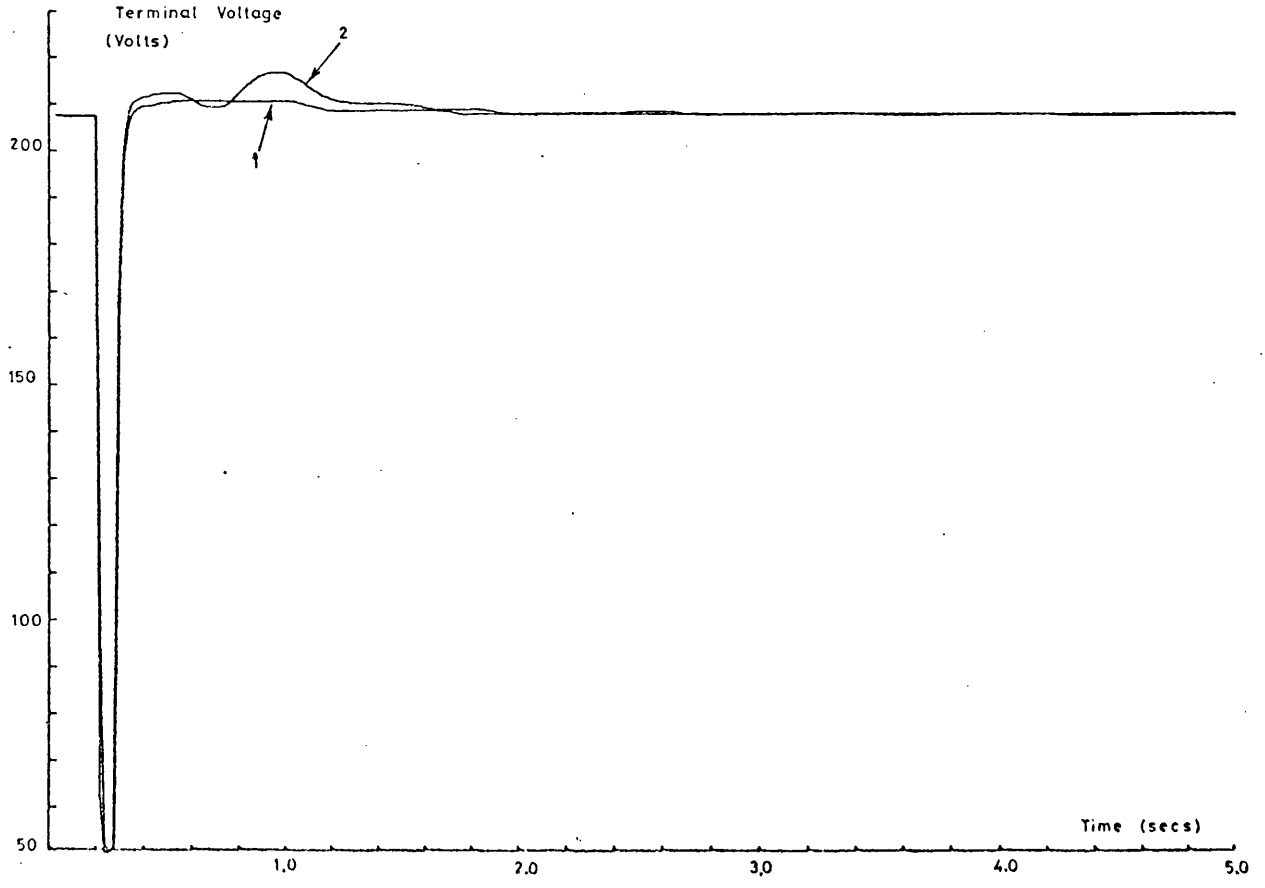


Fig. 8.39 Terminal Voltage against Time

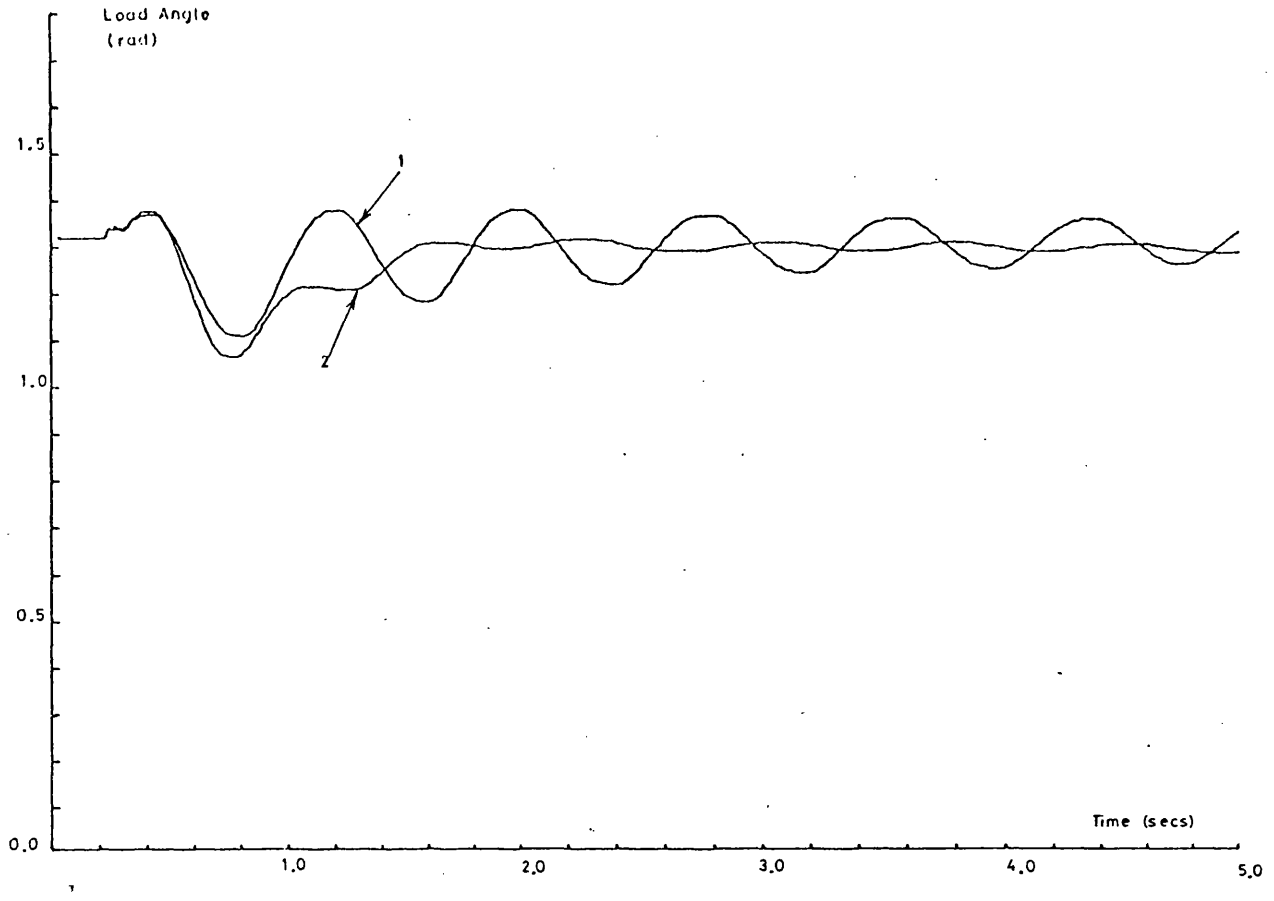


Fig. 8.40 Load Angle against Time

DETAILS OF FIGS. 8.38 , 8.39 & 8.40

Configuration : High gain (as Fig. 4.2)

Plot No. 1 : AVR Control ($V_i = 0$)

Plot No. 2 : AVR plus Digital State Feedback (V_i as eqn. 8.2)

Operating Point : C (Leading VARs — Table 8.1)

Fault Duration : 80 mS

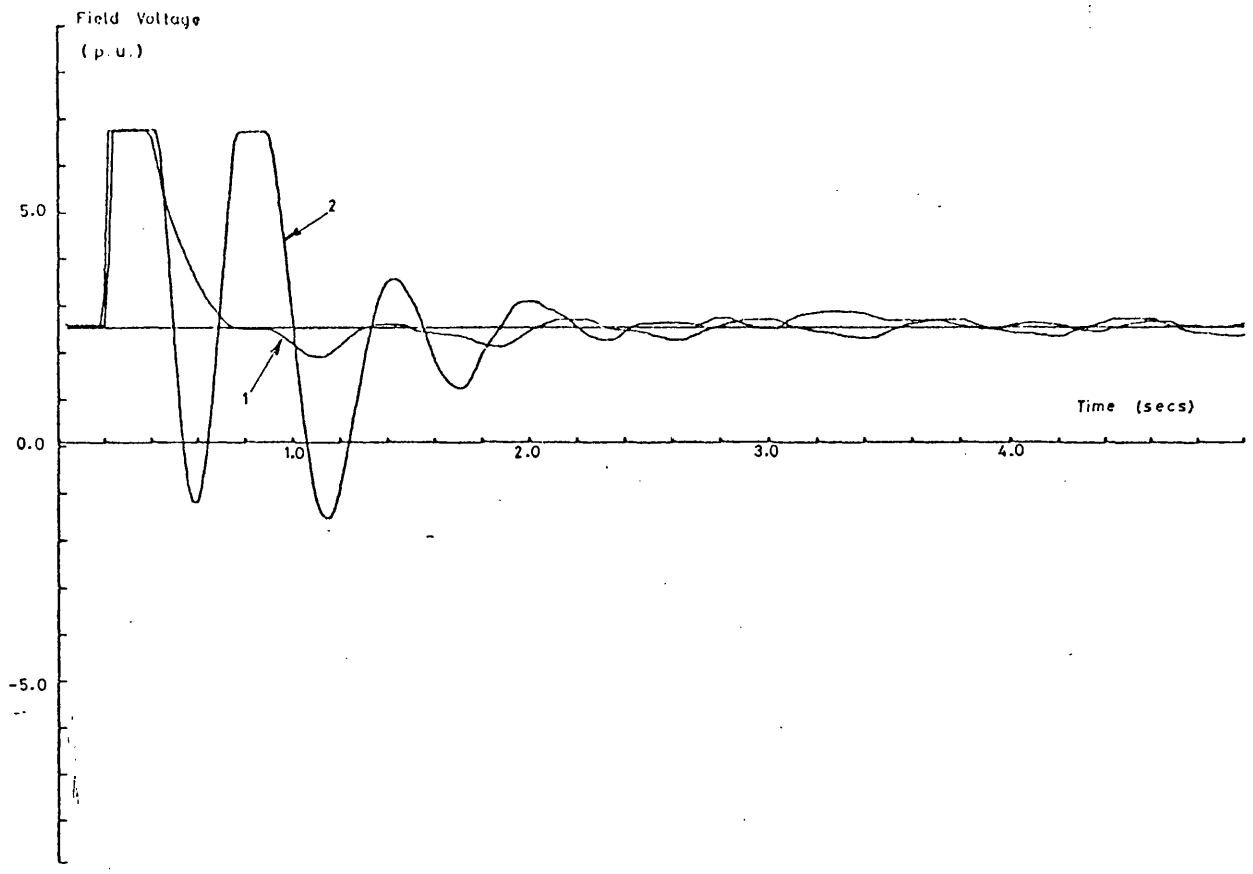


Fig. 8.41 Field Voltage (Excitation) against Time

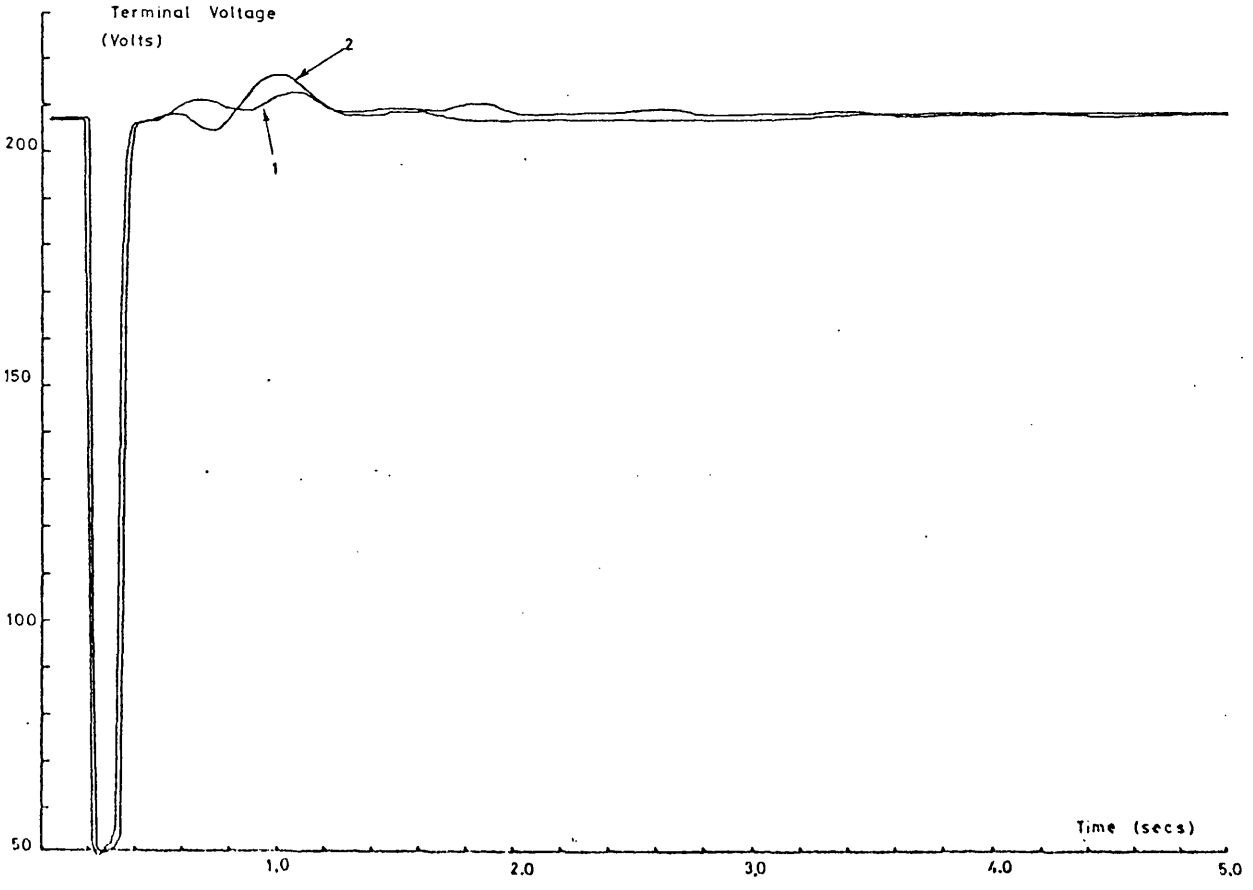


Fig. 8.42 Terminal Voltage against Time

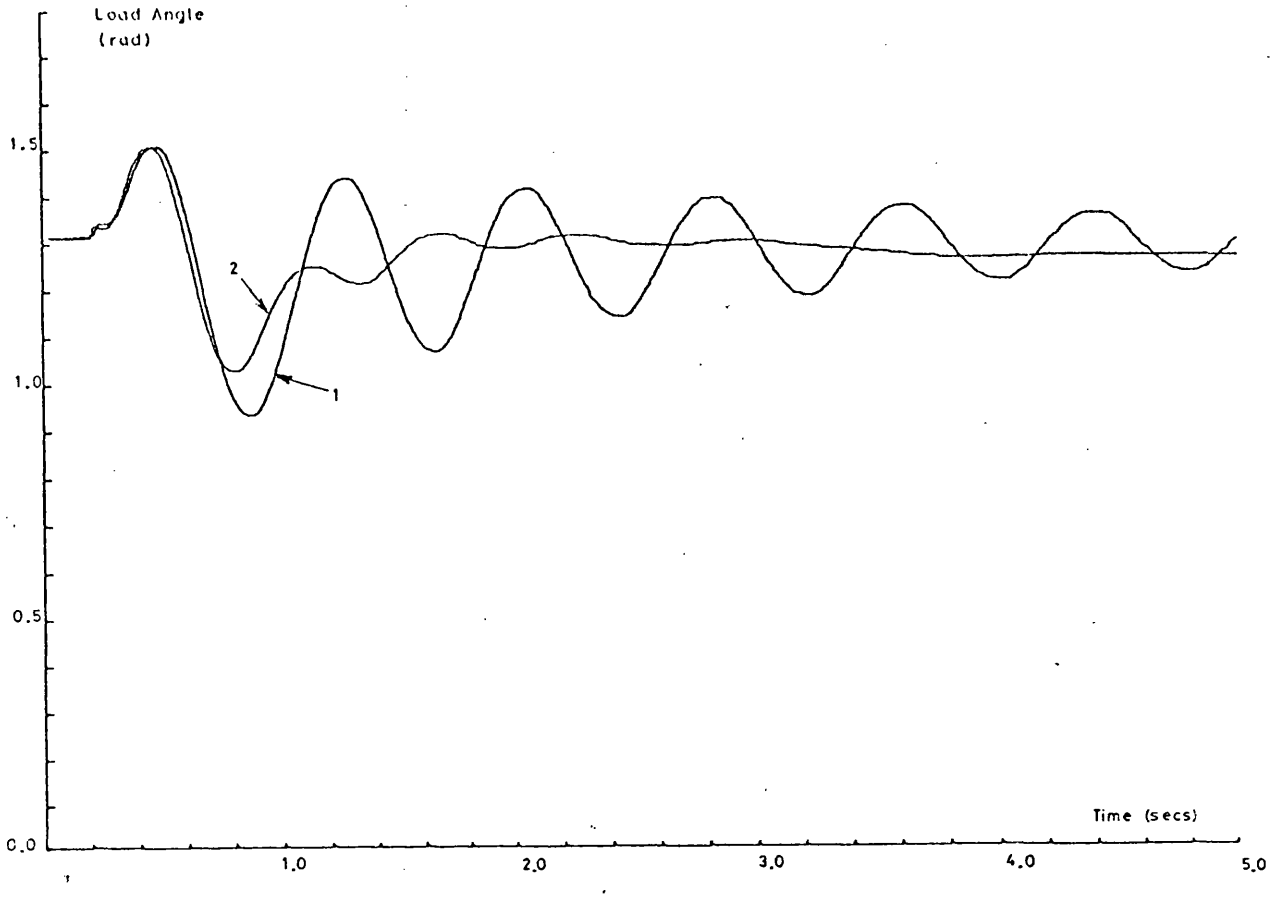


Fig. 8.43 Load Angle against Time

DETAILS OF FIGS. 8.41, 8.42 & 8.43

Configuration: High gain (as Fig. 4.2)

Plot No. 1: AVR Control ($V_i = 0$)

Plot No. 2: AVR plus Digital State Feedback (V_i as eqn. 8.2)

Operating Point: C (Leading VARs — Table 8.1)

Fault Duration: 140 mS

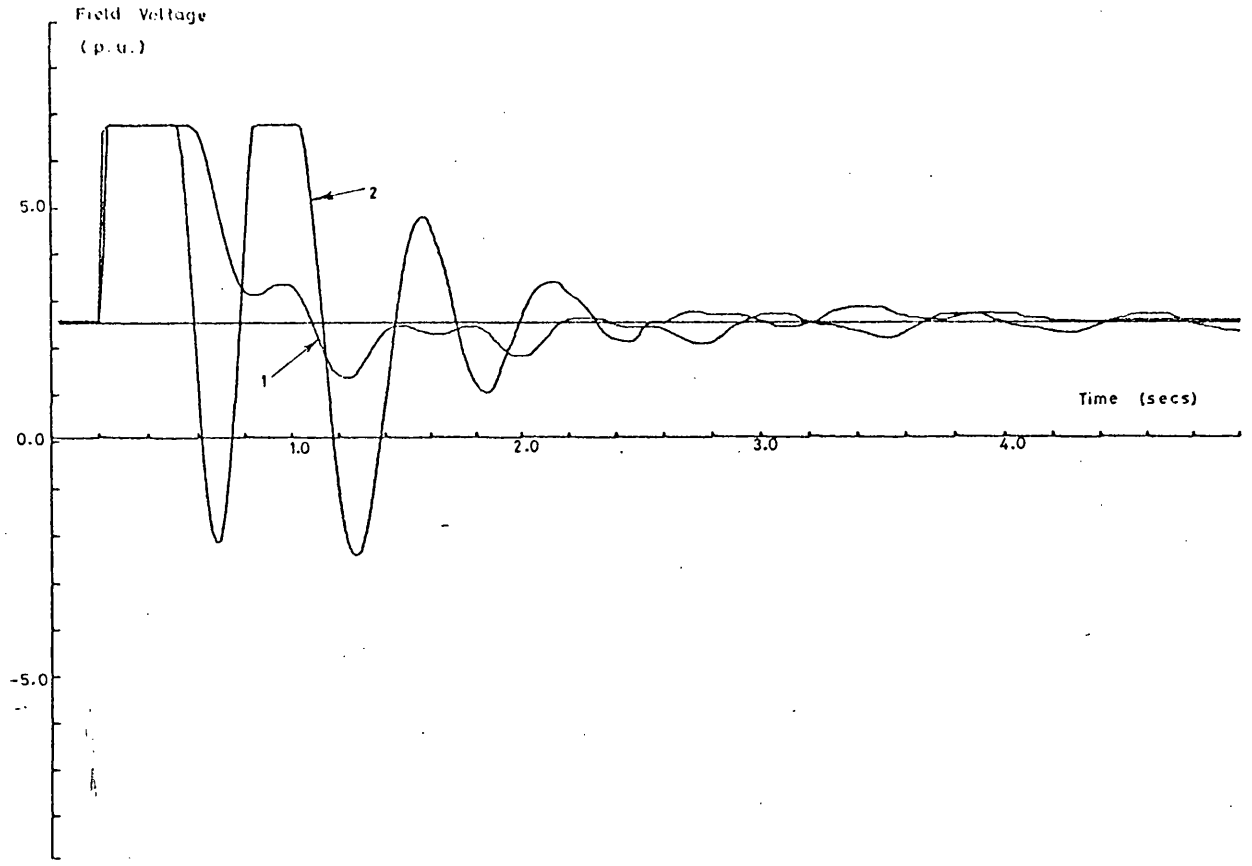


Fig. 8.44 Field Voltage (Excitation) against Time

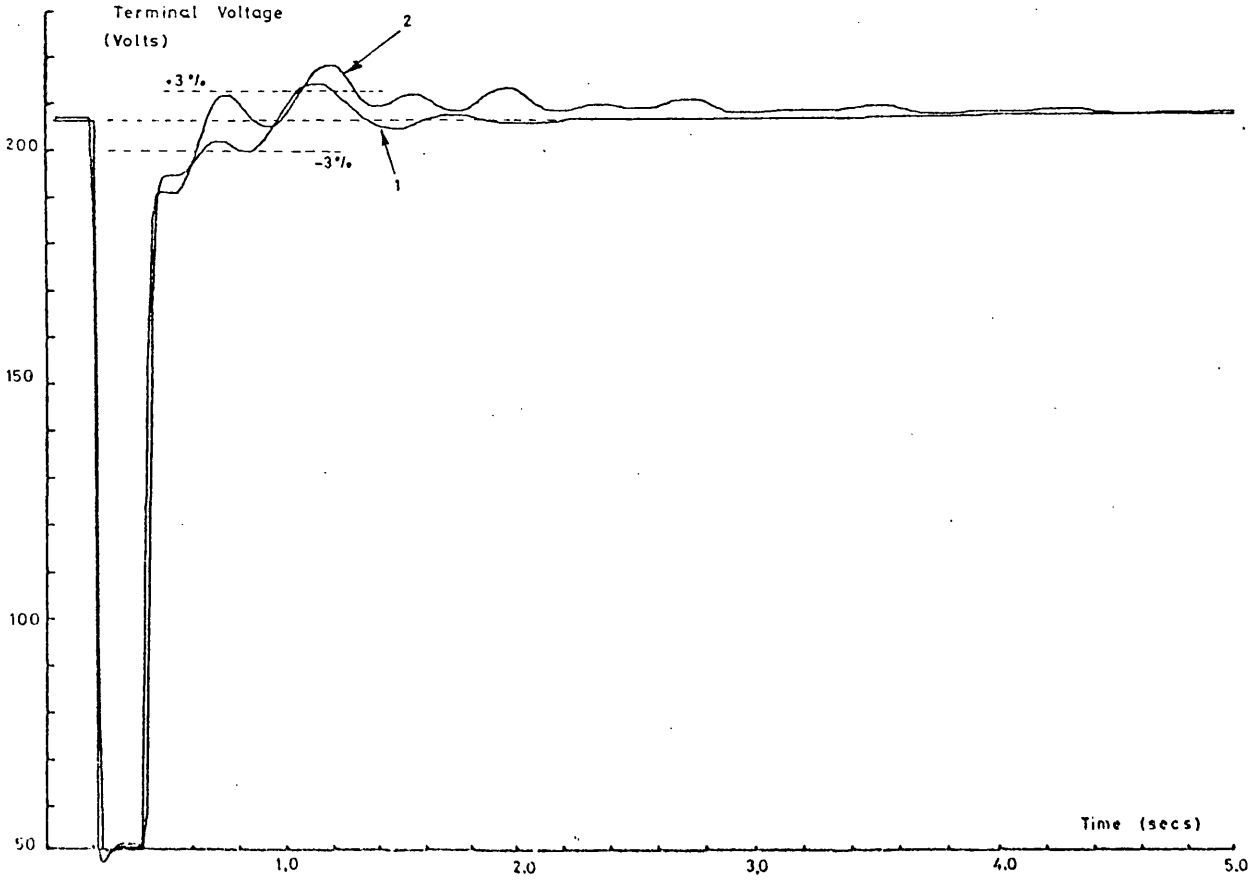


Fig. 8.45 Terminal Voltage against Time

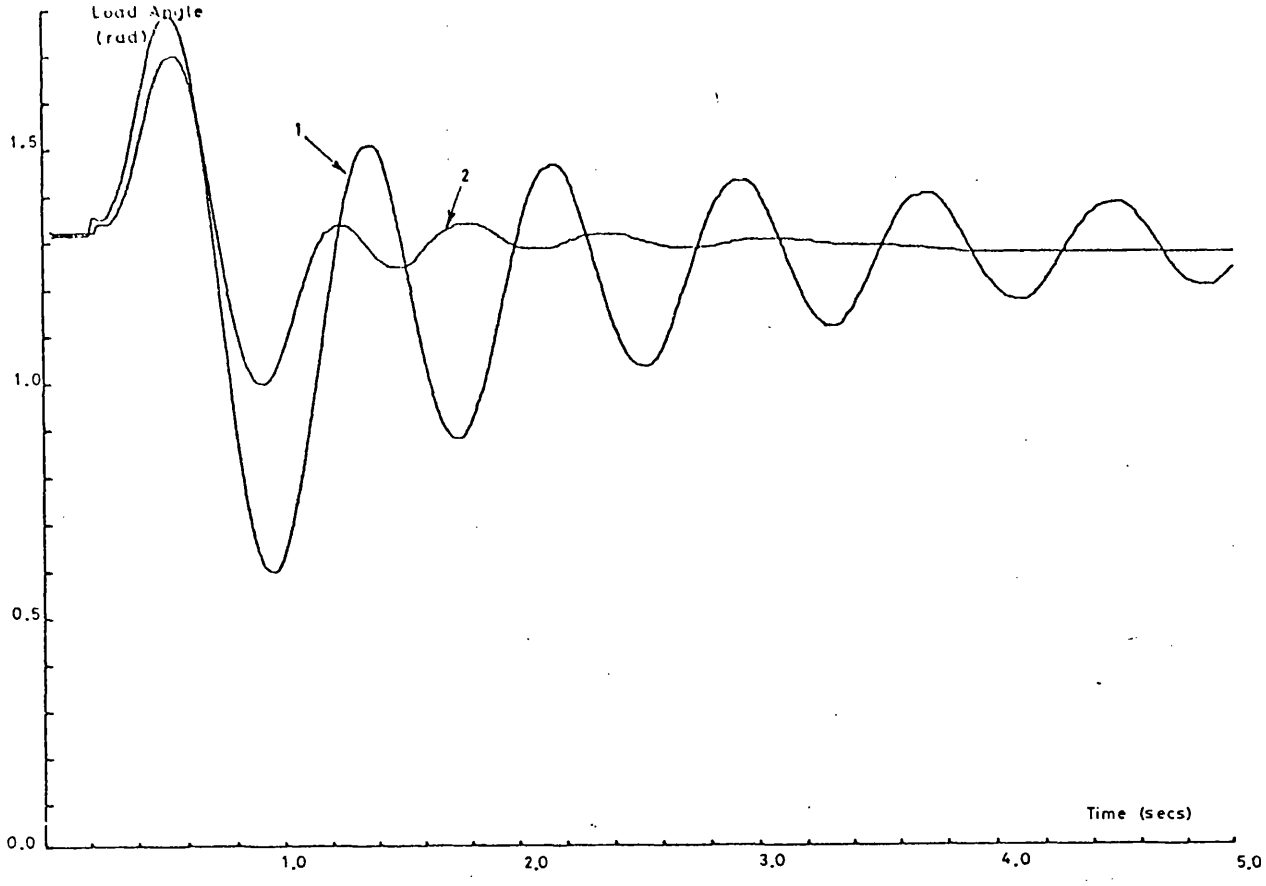


Fig. 8.46 Load Angle against Time

DETAILS OF FIGS. 8.44 , 8.45 & 8.46

Configuration : High gain (as Fig. 4.2)

Plot No. 1 : AVR Control ($V_i = 0$)

Plot No. 2 : AVR plus Digital State Feedback (V_i as eqn. 8.2)

Operating Point : C (Leading VARs — Table 8.1)

Fault Duration : 220 mS

CHAPTER 9

DISCUSSION OF RESULTS

In this chapter, the experimental results presented in Chapter 8 are discussed in general terms.

9.1 ANALOGUE AND DIGITAL CONTROLLERS

The results presented in Figs 8.1 to 8.3 show that a significant correlation in the behaviour of the state feedback controller exists between the analogue and digital implementations. These results show the behaviour of the system when the feedback is applied to the low gain point of the AVR, and although such a form of control is not realisable in a full-size system, it does provide some qualitative indication of the expected behaviour of such a system.

The significant difference between the two feedback characteristics is in the noise content. Fig 8.1 shows that the noise levels in the analogue-derived signal are several orders greater than in the digital equivalent. Attempts in removing the noise from the analogue signal by filtering result in a degradation of the transient response of the controller by introducing an unwanted phase-shift component into the feedback control. This noise content is derived primarily from the transient velocity signal, which is a small percentage of the

relatively high synchronous speed of the turboalternator. It is important that this signal is noise free, not only for the current investigation but also for future studies involving governor control where the time derivative of transient velocity, the rotor acceleration, $p^2\delta$, will be used in the feedback control. Preliminary investigations into the numerical differentiation of the digitally-derived transient velocity signal, shown in Fig 8.19, have provided an acceptably noise free signal representative of rotor acceleration.

In spite of the limitations shown by the analogue state feedback control system, the trends in improved system transient performance predicted by the theoretical studies have been confirmed in the experimental investigations.

9.2 MODERN EXCITATION SYSTEMS

In present systems, the power amplification element of the excitation system is usually either an a.c. exciter or a thyristor exciter. Basically, the former is an a.c. generator with diodes producing an uncontrolled rectification of its output, while the latter produces controlled rectification with the capability of reversal of the field voltage.

The work in the present study has been based on the use of thyristor excitation systems, which allow faster action and field voltage reversal. These characteristics are an essential requirement of the sub-optimal controller derived, as examination of Fig 8.44 shows. A fast-acting exciter is necessary to

force the field voltage to its ceiling value as soon as a fault condition occurs and to follow the required rapidly changing and reversing excitation signal shown in Fig 8.44.

It has been noted previously that the practical implementation of the proposed sub-optimal controller requires that the additional state feedback signals are added to the excitation control signal before the high gain in the forward path of the excitation system. Experimental results on the model system corresponding to this condition have been described in Section 3 of Chapter 8 and these results confirm that the proposed digital excitation controller does lead to significant improvements in the transient performance of the single machine power system. Previous attempts to implement this form of control using analogue transducers have not been successful, mainly because of the combined effects of noise on the state control signals and the high forward path gain. However, a secondary problem was that of backing off the steady state values of certain of the system states with a d.c. level in the analogue controller. Both of these problems have been completely removed by the use of direct digital control.

9.3 EFFECT ON VOLTAGE REGULATION

In general, a synchronous generator system utilising only terminal voltage feedback in a conventional AVR form can be optimised to minimise disturbances to the terminal voltage[2]. Thus, it is intuitively obvious that the inclusion of any

additional feedback signals will, in most cases, detract from the performance of the system in terms of the terminal voltage behaviour. The effect on these additional signals in slightly decreasing the voltage regulation, in return for an improvement in load angle response, must be considered.

Firstly, the additional state feedback signals do not detract from the steady state voltage regulation of the system as the feedback components are derived from signals representative of the difference of these states from the steady state operating point. This is a feature described by American authors as a 'washout' characteristic and is implemented as described in Section 7.4.1. As such, the components of the composite feedback signal are all zero under steady state conditions, with the exception of the terminal voltage signal which behaves in the conventional manner. In practice, it is found to be advantageous if these additional feedback components are not included by the control computer under small disturbance conditions, as this allows for better voltage regulation and provides more computer time for data logging and system updating (eg modelling), as required. Furthermore, such a facility built into the system will naturally be more useful in anticipated future work involving bang-bang type control systems, where it is desirable to revert to a simple linear control in regions near to the desired operating point.

Under transient conditions, when the additional state feedback signals are significant, a slight degradation in the response of the terminal voltage is to be expected. The worst-case

conditions are those of maximum fault duration (220 mS) and leading power factor operation, as shown in Fig 8.45. The significant features of the terminal voltage response are that it should recover to within $\pm$ 3% of its former value within the minimum possible time following the fault. The results of Fig 8.45 show that under the influence of the state feedback controller the recovery of the terminal voltage to these limits is as rapid as that obtained under control of the AVR alone. There is, however, a slightly increased overshoot of voltage, exceeding the 3% limit for approximately 0.1 seconds longer. This characteristic has been deliberately chosen by the form of the weighting coefficients in the performance index of eqn (4.1). The overshoot of terminal voltage for a short period following a fault can be beneficial, provided that it is not of too great a magnitude as to cause over-voltage damage. The station auxilaries will have been underpowered during the fault duration and this temporary rise in voltage can aid their recovery. All the terminal voltage response curves given in Chapter 8 show that this voltage returns to the required steady state value once the oscillations in rotor angle have been completely damped out.

9.4 FIRST SWING STABILITY

In order that the first forward swing of the rotor angle is reduced to the minimum possible value during a fault condition, it is essential that the control law applied does not detract from the field forcing action of the AVR system. Generally,

during a fault condition, it is necessary that the form of control employed forces the field to its maximum positive value for a sufficient length of time to minimise the first forward swing of the rotor. This characteristic is exhibited by the state feedback controller in all the operating regions studied.

9.5 OSCILLATIONS IN LOAD ANGLE

As discussed in early chapters, the use of fast acting excitation systems using terminal voltage feedback alone may result in poor damping. In a poorly damped system, oscillations, once started following a disturbance, will continue causing unacceptable frequency drift for a prolonged period of time. The system will also become more prone to instability following a subsequent disturbance and, for a satisfactory overall performance, it is of prime importance that the oscillations die away quickly.

A general study of the results of Chapter 8 shows that the sub-optimal controller developed meets the above-mentioned requirement, in that oscillations in load angle are damped out in one or two cycles following a major system disturbance. In some instances, for example Fig 8.34, the controller is able to effect a virtual 'dead-beat' return of the load angle to its steady state condition. In other cases, for example Fig 8.43, the constraints of the desired performance are such that a large backswing in load angle must follow the forward

swing. However, in all cases, the damping in subsequent oscillations in rotor angle introduced by the addition of state feedback is always superior to that obtained using terminal voltage acting alone.

9.6 SENSITIVITY

It has previously been noted that the introduction of additional feedback in the form of the system states can, theoretically, lead to significant improvements in the subsequent damping in rotor angle following a large system disturbance. These results of these theoretical studies have been confirmed experimentally by the results given in Chapter 8 of this dissertation.

The necessary state control law has been obtained by off-line theoretical studies of a non-linear mathematical model of the single machine system, using a function minimisation technique, based on dynamic sensitivity functions[16]. Any control obtained from such an analysis must be investigated to establish whether the gain settings of the state feedback signals are sensitive to changes in system parameters and operating conditions. Previous theoretical studies[16] have indicated that, for example, the particular gain settings obtained for the normal full load conditions and a set fault duration of 140 mS remain near optimal for a wide range of fault durations and initial volt-ampere reactance conditions. Experimental investigations have been described in Chapter 8 corresponding to fault durations of

80 mS and 220 mS with initial values of voltampere reactive corresponding to both the maximum and the minimum tappings available on the tap-changer in the model system. These results confirm that the sub-optimal control law is quite insensitive to the given changes in system conditions, and the presence of the extra feedback signals has led to a significant improvement in the transient response of the system, over that obtained with terminal voltage feedback acting alone.

CHAPTER 10

CONCLUSION

The research work described in this thesis has shown that a form of sub-optimal control based on modern control theory can be implemented in a microprocessor with a supervisory computer backup for control of excitation of a model turbo-alternator system. Thus, the results provide an insight into various fields, namely: the excitation control of a synchronous generator, a practical application of modern (sub-)optimal control theory, use of a computer and microprocessor in a real-time environment, development of real-time control systems and programs, design of direct digital transducers for computer control and, finally, development of support facilities necessary for investigations into direct digital control systems.

10.1 ANALOGUE vs DIGITAL CONTROL

Initial investigations in the implementation of the sub-optimal feedback control demonstrated that an unacceptably high noise level was introduced into the system by the use of various forms of analogue transducers for the measurement of certain system variables. In particular, the measurement of the rotor transient velocity presented a severe problem. Generally, measurements of system variables which are time functions (velocity, acceleration, etc) can be made more accurately by

digital techniques than by the equivalent analogue method. This is because of the ability of the digital system to employ timing devices such as high-frequency high-stability crystal clock counters which may measure time with extreme accuracy. In the particular system studied, the use of digital techniques enabled the rotor transient velocity to be measured with great precision and free from noise. This fact is of particular importance in view of the future use which may be made of the time derivative of this signal (rotor acceleration) in further work involving sub-optimal control of the turbine/governor combination.

Further advantages of the digital transducers used in the system have been shown to be their fast response and lack of filtering requirements. An obvious advantage of the direct digital type of transducer in a computer-based control system was also demonstrated by the minimal interfacing requirements for data transfers to and from the control computer (microprocessor).

10.2 LABORATORY INVESTIGATIONS INTO DIGITAL CONTROL

The investigations into the practical implementation of a direct digital controller based on a microprocessor have shown that considerable effort is involved in the development of a laboratory system on which the investigations may be performed. Once developed, however, such a system is extremely flexible and many forms of control may be rapidly implemented. The degree of adaptability and usefulness of the system has been

found to be a function of the layout of the computer hardware system, including the peripheral transducers, and the software system, both on-line and off-line.

The transducers developed in this study incorporate a common form of interface to the microprocessor and may be run in several different modes, or inhibited altogether, under the control of the microprocessor program. The common form of this interface simplifies use of the system in future research, and the program-controlled modes of operation enable different control structures to be formulated without extensive hardware modification.

The real-time control software developed has also been arranged to be modular in concept, enabling incorporation of the various facilities in future control philosophies which may be implemented on the system. Two very powerful support software packages have been written and incorporated into the supervisory computer software system. These packages are the direct-loading cross-assembler and the microprocessor on-line debugging technique. They function through the hardware facilities provided by the microprocessor to computer interface and have proven invaluable in rapid development and testing of microprocessor control programs. Another important feature of this interface was the facility thereby provided for storage of microprocessor programs on the magnetic tape and disk units of the supervisory computer and the rapid reloading of programs on demand. This eliminated the major time-consuming task in microprocessor program development of program regeneration and

reloading.

10.3 SUB-OPTIMAL CONTROL OF THE MODEL SYSTEM

In the past decade, many publications have appeared advocating various modifications to the basic form of the most widely used excitation system. The designs have mainly been based on intuitive methods, linear system optimisation and non-linear optimisation of computer models. Little consideration, by comparison, has been given to the application of modern control techniques to practical systems. It is only recently that authors have commenced to apply the ideas and concepts of modern control theory to practical turbo-alternator systems[24,50]. The research covered in this thesis has attempted to apply the methods of optimal control in a practically viable manner by using a form of sub-optimal control based on the linear feedback of some measured state variables.

The study has shown that a feedback of load angle, rotor transient velocity and voltage behind transient reactance can produce a significant improvement in the response of the system following a major disturbance. Under the control of the computer-generated feedback law, the system has been shown to be inherently more stable, as evidenced by the controlled behaviour of the load angle oscillations. The control applied has exhibited minimal degradation of the response of the terminal voltage in achieving this characteristic and, as discussed, the overshoot produced in the voltage may be benefi-

cial to the recovery of the station auxilaries.

10.4 SUGGESTIONS FOR FURTHER WORK

10.4.1 Excitation Control

The results obtained in this study have shown that the excitation input to the synchronous generator must rise and fall very rapidly to achieve the sub-optimal control function and during the large disturbance conditions tends to approximate to a bang-bang control function. The bang-bang control system has certain advantages over the linear feedback system in that the controlled system errors and derivatives of error can usually be reduced to zero in a shorter time due to the maximum effort form of input to the system. Some simple preliminary investigations using a bang-bang form of control function were made[51]. In these, the input was limited to maximum positive, maximum negative or nominal steady state levels and optimised by means of varying the switching times to reduce a performance index. These simple tests indicated that some improvement in response can be made.

A bang-bang form of control is more amenable to implementation on a digital control system due to the discrete nature of its behaviour. Early investigations into the action of contactor servo-mechanisms[52] and, more recently, the application of the state-transition method of modern control theory[53], have shown that the system error and derivates of error of a linear nth order system may be reduced to zero in minimum time by the

application of (n-1) input switching states. Further, Tou[53] has shown that an increased number of switching states may be required for optimal control of non-linear discrete-data systems, dependence being placed on the non-linearities present and the rather subjective criteria defining optimum performance. Engineering intuition would predict that a form of sub-optimal bang-bang controller for the single-machine synchronous generator system would involve typically two or four switchings of full input excitation as third and fifth order models may be used to form a reasonable linear representation of the system. This presupposes that some form of 'dead-band' is allocated around the steady state operating point of the practical non-linear system, such that 'chattering' (continual high-speed switching) does not occur. Within this dead-band, a simple form of linear feedback could be employed and, in the particular case of the synchronous generator system, AVR feedback alone would probably be sufficient.

Various theoretical studies have been performed to determine the switching hypersurface (in state space) of linear time-invariant systems[54-56] but less work has been done in consideration of non-linear and time-invariant systems, particularly in relation to practical systems[57,58]. There are three major methods of determining the switching hypersurface: off-line analytical studies, on-line heuristic learning systems and some forms of model adaptive switching controller. The first of these methods already forms the basis of further research within the group[44] and will not be commented on here in any more detail.

Some preliminary investigations into the heuristic learning method of Waltz and Fu[59] and later developments[60] were performed on a digitally-simulated generator system. The results suggested that a learning control system having no a priori information about the system must have a slow learning rate in order that the algorithm is stable. Even if the control system is supplied with a basic knowledge of the system, the learning rate restriction still applies. In the practical case of the synchronous generator system, where certain conditions such as short-circuits occur infrequently, a slow rate of learning is not acceptable and further investigations along these lines were discontinued.

It would seem that a form of model adaptive switching controller holds some promise for future investigations. This basic philosophy of this technique is that of using a very simplified model of the system with a continual adjustment of the parameters to force a best fit to the observed system behaviour at the current operating point. Errors due to the model simplification are kept small by the self-checking or learning abilities of the model. With the large reserves of memory available in the digital control system, it would appear possible to store a range of models to be used in appropriate operating regions, giving a total effect of accurately reproducing the non-linearities of the system and yet retaining simple algorithms at each stage of the control function. A large amount of literature is available on adaptive control techniques[61] and it is considered that further investigations into the application of these philosophies

to the synchronous generator system would prove fruitful.

10.4.2 Combined Governing and Excitation Control

The results presented here have also shown that the use of a sub-optimal control philosophy for excitation control of a synchronous generator can increase the transient stability limit particularly when used in conjunction with a fast-acting excitation system. The implementation of such a control increases the range of operating conditions in which a longer time is available following a disturbance for the prime-mover input power to be adjusted to restore the power balance condition. This gives the governor a more significant role in the consideration of synchronous power system transient stability.

The form of governor used on steam turbo-alternators has changed very little since its original introduction and is modelled on the velocity governor believed to have been invented by James Watt for use on early reciprocating steam engines. Being mechanical in nature, the valve responds relatively slowly (with respect to the electrical signals) to a command signal, thus producing a lag in the system response. Most large turbo-alternators also include reheating of the steam during the turbine cycle and this introduces further lags due to the nature of the response of the entrained steam within the system. The effect of these lags in response is such that the governor is not expected to have much effect on the first swing stability of the system, but may be significant

in damping out further oscillations.

To date, little practical investigation has been made into governor action, although some note has been made of the effect of the governor[62] and theoretical studies based on linearised small deviation techniques have been made[63,64]. Such studies are not strictly applicable to the behaviour of the system under large disturbance conditions, but useful inferences can be made with careful reservations. These inferences are generally that shaft velocity and acceleration signals are predominant in the formulation of the 'best' feedback control law. Obviously, some velocity signal must be used to control power input to cope with conditions of slowly changing load. In theoretical studies under large disturbance conditions, it is reported[65] that a governor signal based on acceleration ($p^2\delta$) feedback has the effect greatly increasing the apparent machine inertia. This reduces the speed changes during a fault transient and, except under long fault clearance conditions, produces a more stable system. Thus, it would appear likely that some form of governor comprising a compound of velocity and acceleration feedback signals could provide a further improvement in the dynamic stability of the synchronous generator system.

An analysis has been performed for a theoretically-derived sub-optimal controller for the governor and excitation system combined[66] and it is considered that a useful continuation of the present work would be to extend this for control of the practical micro-machine system in a manner to that performed for the excitation system alone. As previously mentioned, the

hardware and software of the laboratory-based digital control system have been specifically designed with this future work in mind. It is anticipated that such work will eventually lead to sub-optimal excitation control and governing of a two-machine system closely representative of the practical case.

10.4.3 System Modelling

Experience gained with the microprocessor in the laboratory model real-time control system has lead to the conclusion that microprocessors could be usefully employed to generate some of the functions of the system being modelled to effect an improvement in the simulation. Such an improvement would be provided by the programmable nature of the microprocessor which would enable the system characteristics to be rapidly and simply changed. Microprocessors are becoming increasingly more useful in this capacity in view of their diminishing cost and are particularly attractive in a system with a mainframe computer because of the program assembly, loading and testing facilities made available by the techniques described elsewhere in this thesis.

In order to test the effectiveness of the microprocessor in generating the necessary functions, the AVR of Fig 4.2 was programmed into the I8080 microprocessor which had previously been performing the state feedback function for excitation control. The generation of the transfer function was performed by transforming the AVR characteristics into a set of difference equations

in a manner analogous to that described in Section 7.4.1. These equations were then programmed into the I8080 in the standard mnemonic form. The operation of the digital AVR so created was virtually indistinguishable from the behaviour of the analogue model and, as the work was intended as an initial feasibility study, no results have been included here. However, some useful information was gained in that the digital AVR occupied 600 bytes of microprocessor memory and needed 3.0 mS in every 20 mS cycle to perform the calculations necessary for the simulation. It is anticipated that future work will include the provision of a hardware multiplication facility to the microprocessor and this should increase the speed by a factor of between 5 and 10 for this simulation.

This brief study indicated the usefulness of the microprocessor for function simulation and it is anticipated that future work will involve the microprocessor-based simulation of a complete turbine/governor combination either in a separate microprocessor or as part of the system contained within the microprocessor currently performing the excitation control function.

10.4.4 Real-Time Monitor and System Software

The software system described in Chapter 7 provides an efficient and versatile real-time control and data-logging facility. However, certain modifications and developments could be made during the course of further research on the facility to

improve both the control function and the ease of program development.

The most obvious development of the system is the extension of the interface device driver facility to extend the capabilities of the PDP-11 for multi-microprocessor data transfers. This would enable the computer to control and communicate with the four microprocessors which may eventually be used in the multi-machine generation system. This development should be fairly readily performed as system developments to date have been made with this future extension in mind. It is also anticipated that the full foreground/background programming capability of the system will be developed to allow such operations as real-time control, data-logging, graph plotting and program development to be performed concurrently as needs dictate.

Both the cross-assembler (MICRO) and the on-line debugging technique (DEBUG) have been written with future extensions in mind. MICRO has been developed such that its internal tabular structure of mnemonic/machine code instructions may be readily changed to allow for future work with different microprocessor types. Furthermore, the assembler recognises macro and conditional assembly expressions and also complex argument types and has provision for these expansions to be 'slotted in'. Such improvements would greatly aid the development of more sophisticated control programs for future work.

10.5 ACKNOWLEDGEMENTS

During the course of the research, the guidance and counsel of my supervisor, Mr A R Daniels, is gratefully acknowledged. The work was performed as a part of Mr Daniels' overall research into power system performance at the University of Bath, and, in this connection, I wish to thank the School of Electrical Engineering, particularly the then Head of School, Professor A J Eales, for making the facilities available.

I would like to thank Mr G S Chana for his invaluable assistance in the off-line optimisation studies and in discussions on some of the theoretical problems involved. I would also like to thank Dr B A White for his discussions and practical assistance during our mutual shared initial investigations into microprocessor systems. Thanks are due also to Mr V S Gott and Mr P A Hazell for practical assistance during the experimental tests.

Lastly, thanks are given to the Science Research Council for financial assistance and to Lucas Aerospace Ltd for both financial support and, more valuably, leave of absence, without which my involvement in this research would not have been possible.

APPENDIX

Note

1. The Appendix has been written as a separate volume to the main body of the thesis and thus may appear to contain conflicting diagram and section numbers. All references to diagrams and sections made within the Appendix refer to the Appendix unless specified otherwise.
2. Throughout the Appendix, zero has been represented by the character Ø to avoid confusion with the alphabetic letter O.

SECTION 1

INTRODUCTION

The digital control system for the micromachine model currently (March 1977) contains two processing units: a PDP-11/20 computer and an I8080 microprocessor. The function of the PDP-11/20 computer is to perform the main supervisory functions during the real-time control operation and also to provide high level program development and storage facilities for both its own programs and those of the microprocessor. During the control operation, the microprocessor acts as a front end data collecting and filtering unit and implements the control function under the supervision of the main PDP-11 computer. The microprocessor is situated physically close to the system under control to reduce transducer interconnections and also to provide immediate local control in the event of certain types of system failure. The main PDP-11 computer is situated in an adjacent laboratory, but could equally well be in any remote location, and communicates with the microprocessor through a transmission link. A general schematic diagram of the system is given in Fig 1.1.

The transducers measuring and controlling the micromachine variables all communicate directly with the microprocessor, and thus, indirectly, with the PDP-11 computer. As described in the following sections, an attempt has been made to design all transducers to function in a direct digital mode to increase system efficiency and reliability, The peripheral units which normally communicate with the PDP-11 computer directly (magnetic tape, magnetic disk, line printer, etc) are retained in this mode. They are utilised for high level communication and data storage, both during the program development phase and during the real-time control operation.

The design of the electronic hardware and program software for the system has been performed in a manner which allows the I8080 microprocessor to be considered as a fully integrated part of the PDP-11 system. As such, it can be regarded as a standard

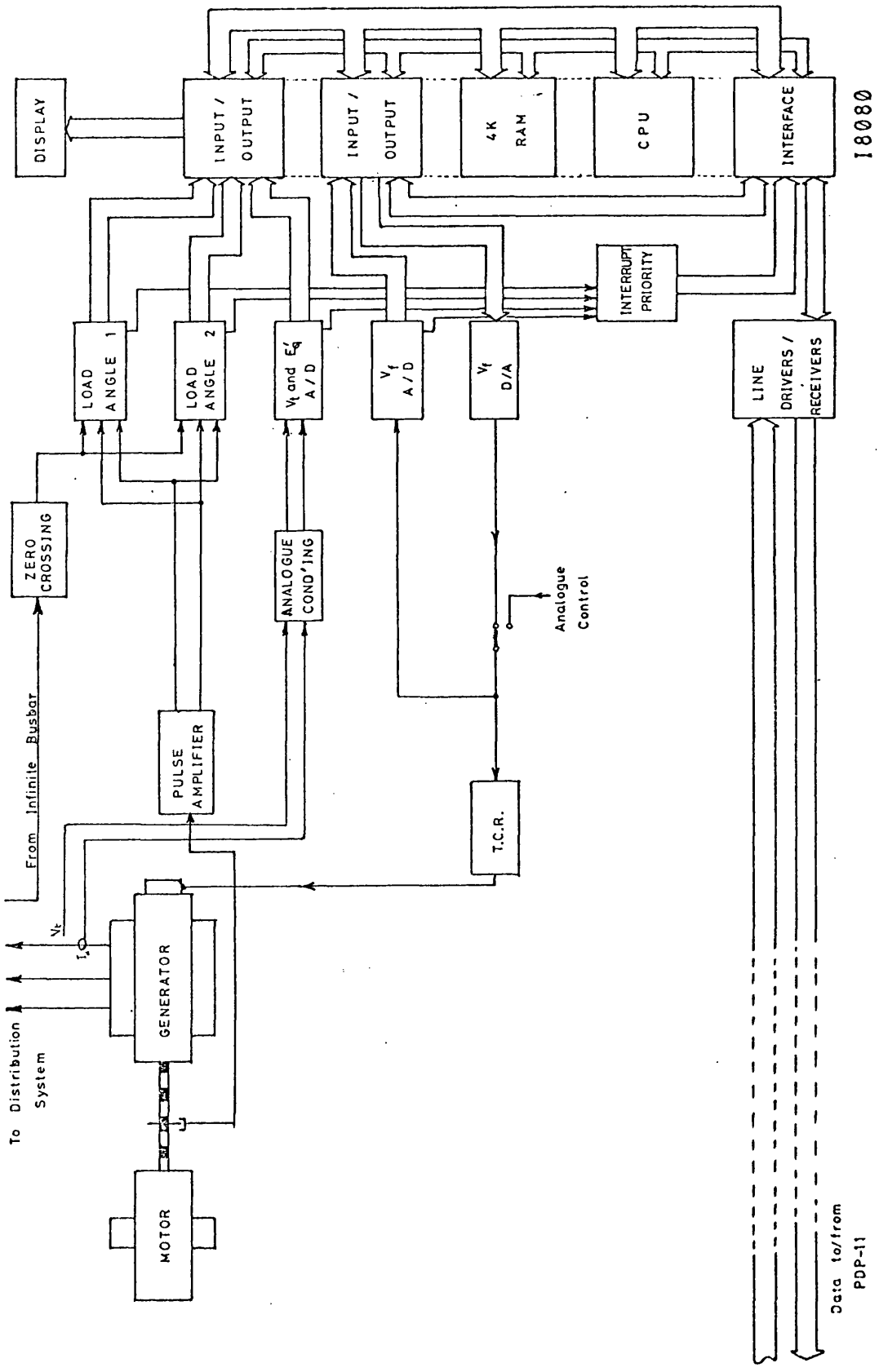


Fig. 1.1 Digital Control System (Simplified Schematic)

peripheral of the system and functions with all the advantages of device-independent program operation. Basically, this means that program and/or data information being transferred to or from the microprocessor can be transferred via, or stored on, any suitable PDP-11 peripheral device. This operation can be performed using standard system programs and command modes. The system thus produced is an extremely flexible one, in which operational program changes are easily implemented, yet simplicity of operation is retained by the standardised nature of the commands and transfers.

This appendix describes, in detail, the hardware components of the digital real-time control system which have been designed, developed and constructed by the author. Details of other components of the system will be found in the relevant manuals listed in the References. The details of the software system are also recorded in this appendix in a similar manner.

SECTION 2

PDP-11/20 to I8080 INTERFACE CARD

The interface circuitry was primarily designed to allow interrupt and data-transfer facilities between a PDP-11/20 computer and the I8080 microprocessor system. However, the standardised nature of the interface input/output ports will grant compatability with many other systems and these are inferred in the following text whenever the PDP-11/20 computer is referred to. The circuitry is such that simple programming switches and display lights may be connected to the interface, via suitable buffers, to form a functional stand-alone microprocessor system.

The input/output ports of the Interface Card may either be connected directly to the direct digital interface (DR11-A) of the PDP-11/20 computer or may be used to drive a data link with suitable buffering to connect these two systems.

2.1 GENERAL FUNCTIONAL DESCRIPTION

This section describes the operation of the Interface Card in general functional terms and is divided into four sections which relate to the four major functions of the device:

1. Programming and Direct Memory Access
   This facility allows programs and/or data to be loaded into the microprocessor memory directly from the PDP-11/20 computer.

2. Control and De-bugging
   This facility allows the microprocessor to be run under the direct control of the PDP-11/20 computer for program testing and de-bugging purposes.

3. Interrupt
   The interrupt facility allows the microprocessor to both interrupt and be interrupted by the PDP-11/20 computer

for data-transfer and program-flow control purposes.

4. Data Transfer
   A facility to allow the transfer of 8-bit data words under program control between the I8Ø8Ø and PDP-11/2Ø in both directions

2.1.1 Programming and Direct Memory Access Operations

In order to allow an external system to write data (or programs) into a series of microprocessor memory locations, or to read those locations, it is necessary that the microprocessor CPU relinquishes its hold of the system data, address and control busses and that the external device has the ability (via the interface) to control these busses. The Interface Card provides this facility by decoding a certain signal from the external device and subsequently forcing the I8Ø8Ø CPU into a HOLD state thus enabling the bus-drivers in the interface to gain control of the required busses.

Basically, the Interface Card generates three sets of signals during Direct Memory Access (DMA) operations. The first set of signals is derived from the interface's own program counter and these are used to provide the addressing information. The second set forms the data-transfer control signals and these are largely generated within the Interface Card and applied to its own buffers and to the microprocessor memory card. The third set of signals comprises the data being transferred and is obtained from the external signal and applied to memory in the case of a WRITE operation, or is obtained from the memory and transmitted to the external system in the case of a READ operation.

2.1.2 Control and De-bugging Operations

To facilitate de-bugging of the microprocessor programs and also to allow certain control functions such as resetting of the program counter to re-start a program, some of the system con-

trols are made available to the PDP-11/2Ø computer via the Interface Card. Variously, these allow the computer to re-start a microprocessor program as previously mentioned, to cause the microprocessor to enter a WAIT state while its busses and status registers are examined and also to cause the microprocessor to run a program in 'single-step' mode under computer control while the computer software is performing certain functional checks of the microprocessor program flow and CPU status.

2.1.3 Interrupt Operations

As the I8Ø8Ø has only a single level of interrupt priority, the interrupt priority servicing has to be performed in a hardware stack. The Interface Card provides the base (or first) priority level which it allocates to the PDP-11/2Ø computer. It also provides the facility for additional interrupt hardware to be connected in order of decreasing priority and generates the master control signals for these additional interrupt stages (Section 7). The Interface Card also causes an interrupt to be sent to the PDP-11/2Ø whenever data is loaded into one of the microprocessor output ports (normally Port Ø).

2.1.4 Data Transfer Operations

To avoid loss of data or wasted transmission time, all data transfers are performed by the 'hand-shake' technique, i.e. both data transmission and data receiving are acknowledged by flags. This facility is provided in both directions by the Interface Card which contains several 'flip-flops' to record the state of the data transfers.

2.2 INTERFACE CONTROL FORMAT

2.2.1 General Description

The Interface Card has been designed such that all the previously defined functions of the interface and microprocessor system can

be achieved using two uni-directional 16-bit data highways. This has necessitated a moderately complex coding and multiplexing system for some of the desired functions and also a limitation on the size of the data portion of the transmitted word to 8 bits. As the latter is, however, the maximum byte size which can be handled by a single instruction of the I8Ø8Ø, it is not a great restriction.

2.2.2 Interface Input Word

The Interface Input Word is the 16-bit input to the interface, either from the PDP-11/2Ø computer or from any other compatible system. It is physically presented to the Interface Card as signal lines OUT Ø and OUT 15 on socket J2 (pins 5 to 2Ø).

The 16-bit word is structured in three sections as shown in Fig 2.1. Bits Ø to 7 form the DATA portion of the word (although these may at various times represent program code, address information, etc, depending on the mode). Bits 8 to 11 form the MODE portion of the control word and bits 12 to 15 form the function portion. The role of each of these sections is explained in greater detail below.

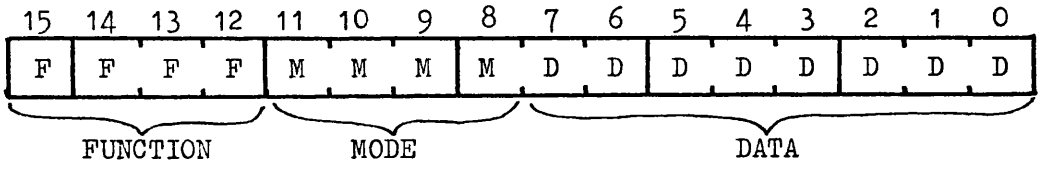


Fig. 2.1 Interface Input Word Format

2.2.3 Interface Output Word

The Interface Output Word is a 16-bit word available on signal lines IN Ø to IN 15 on socket J1 (pins 5 to 2Ø). It contains either information returned via the interface as a result of a request made in the structure of the input word or it may contain

information which has been generated for transmission by the I8Ø8Ø CPU. The nature of the contents is clear from the status of the interrupt and data-transfer flip-flops and the mode of the previous input word.

2.2.4 Auxiliary Controls

The Interface Card has two additional external inputs, NEW DATA RDY and DATA TMTD (socket J2, pins 4 and 3). These are both pulses of logic level '1' for a duration of 5ØØnS and respectively signify the transmission of a new data word by the PDP-11/2Ø (or other system) and the reading of a data word on the DR11-A input of the computer. The use of these signals for 'hand-shaking' in data transmission is explained in greater detail later.

The two additional outputs of the Interface Card are REQ A and REQ B (socket J1, pins 4 and 3). When the Interface Card is connected to the PDP-11/2Ø computer, the REQ B signal indicates that the microprocessor has generated an interrupt of the computer and the computer will respond according to its program. The REQ A signal acts as a DONE signal to the PDP-11/2Ø to signify that there are no further data transfers in progress. Both these functions are explained in greater detail in Sections 2.3.3 and 2.3.4.

2.2.5 Interface Control

Control of the interface, and thus the microprocessor, is achieved by means of the bit pattern contained within the Interface Input Word. The bit pattern of lines OUT 8 to OUT 11 specifies the MODE of control exercised and it is permissible that certain modes may be engaged concurrently, as is the case of the HOLD and LOAD ADDR modes, while other modes may cause an interface or CPU conflict. In order to keep the circuitry of the Interface Card as simple as possible, a limited hardware check has been made on the validity of the Input Word such that modes of operation other than those listed below may cause spurious results. A more extensive software check could be incorporated in the Device Handler for the DR11-A if the user so requires.

RUN mode

F	F	F	F	Ø	Ø	Ø	Ø	D	D	D	D	D	D	D	D

The normal RUN mode of the microprocessor system is specified by the bits OUT 8 to OUT 11 being set to logical 'Ø'. If the bits OUT 12 to OUT 15 are also set to logical 'Ø', then the interface exercises no control over the running of the microprocessor and the state of bits OUT Ø to OUT 7 is irrelevant. Otherwise, the significance of the bits is as follows:

OUT Ø ----- OUT 7	DATA or INT INST	Either an 8-bit data word or an 8-bit interrupt instruction depending on the state of OUT 14 and OUT 15 as below.
OUT 8	RESET	Logical 'Ø'
OUT 9	HOLD	Logical 'Ø'
OUT 1Ø	WAIT	Logical 'Ø'
OUT 11	LOAD ADDR	Logical 'Ø'
OUT 12		Not used
OUT 13		Not used
OUT 14	DATA	A logical '1' signifies that the bits OUT Ø to OUT 7 contain data for transfer to the microprocessor. A logical 'Ø' appears at the DONE flag of the microprocessor (normally Input Port 1, bit 7) indicating that data is being transmitted. No interrupt is generated. A logical 'Ø' also appears at REQ A which acts as the DONE flag to the PDP-11/2Ø. The interface directs the data to microprocessor Input Port Ø and if this port is subsequently read, all flags are restored to logical '1'. It is not necessary to clear OUT 14 before reloading the next dataset.
OUT 15	INTERRUPT	A logical '1' will cause an INT REQ to be generated in the microprocessor

and a logical 'Ø'appears at REQ A. If the microprocessor has been interrupt enabled it will interpret OUT Ø to OUT 7 as a single-byte instruction and branch accordingly. REQ A is restored to '1' when the instruction has been read and normal processor operation continues subject to the contents of the interrupt subroutine.

RESET mode

F	F	F	F	Ø	Ø	Ø	1	D	D	D	D	D	D	D	D

OUT Ø ----- OUT 7		Not used
OUT 8	RESET	A logical '1' will cause a reset pulse of 2 microseconds duration to be sent to the $\overline{\text{RESET}}$ input. This restores the processor's internal program counter and instruction register to zero. Program execution will recommence at memory location zero. The interface program counter and flags are also reset.
OUT 9	HOLD	Logical 'Ø'
OUT 10	WAIT	Logical 'Ø'
OUT 11	LOAD ADDR	Logical 'Ø'
OUT 12		Not used
OUT 13		Not used
OUT 14		Not used
OUT 15		Not used

HOLD mode

F	F	F	F	Ø	Ø	1	Ø	D	D	D	D	D	D	D	D

OUT Ø ----- OUT 7	May not be of significance depending on the FUNCTION, see below

OUT 8	RESET	Logical 'Ø'
OUT 9	HOLD	A logical '1' will cause a HOLD REQUEST signal to be sent to the microprocessor. The interface circuitry clocks the signal to coincide with the internal ϕ_1 clock and causes a HOLD state to occur for DMA or direct I/O control within approximately 3 microseconds. The bit must be held high during the total DMA interval.
OUT 1Ø	WAIT	Logical 'Ø'
OUT 11	LOAD ADDR	Logical 'Ø'
OUT 12	$\overline{\text{MEMORY}}$/ I/O PORT	A logical 'Ø' will enable a DMA of the memory location whose address is currently stored in the interface address buffer. A logical '1' will enable the corresponding input/output port to be accessed.
OUT 13	$\overline{\text{READ}}$/WRITE (If OUT 12 = 'Ø')	A logical '1' will cause contents of OUT Ø to OUT 7 to be written into the microprocessor memory location currently referenced by the interface address buffer. A logical 'Ø' will cause the same location to be read. No flags are set. Data read appears at IN Ø to IN 7.
	$\overline{\text{INPUT}}$/OUTPUT (If OUT 12 = '1')	A logical '1' will cause the contents of OUT Ø to OUT 7 to be output to the output port whose address is currently stored in the high-order byte of the interface address buffer. A logical 'Ø' will cause the contents of the corresponding input port to be obtained and made available at IN Ø to IN 7. No flags are set.

OUT 14	AUTO ADDR	A logical '1' will cause the interface address buffer to be automatically incremented or decremented (depending on OUT 15) following each DMA operation.
OUT 15	$\overline{\text{INC}}$/DEC	In the auto inc/dec mode (OUT 14 = '1') a logical 'Ø' will cause the address buffer to be automatically incremented and a logical '1' will cause it to be automatically decremented following each DMA operation.

WAIT mode

F	F	F	F	Ø	1	Ø	Ø	D	D	D	D	D	D	D	D

OUT Ø ----- OUT 7		Not used
OUT 8	RESET	Logical 'Ø'
OUT 9	HOLD	Logical 'Ø'
OUT 10	WAIT	A logical '1' will cause the microprocessor to enter a WAIT (T_W) state for as long as the bit value remains high.
OUT 11	LOAD ADDR	Logical 'Ø'
OUT 12		Not used
OUT 13		Not used
OUT 14	DISP ADDR	A logical '1' will cause the contents of the microprocessor memory address bus to be available at IN Ø to IN 15 during the WAIT state: IN Ø — Memory Address Bit Ø IN 1 — Memory Address Bit 1 ---- — ---------- IN 15 — Memory Address Bit 15 A logical 'Ø' will cause the data from memory to be displayed in bits IN Ø to IN 7 and the CPU

status to be displayed in bits IN 8 to IN 15 as follows:

IN Ø	Data Bit Ø
IN 1	Data Bit 1
----	----------
IN 7	Data Bit 7
IN 8	INTERRUPT CYCLE
IN 9	MEM WRITE CYCLE
IN 1Ø	STACK CYCLE
IN 11	HALT
IN 12	I/O INPUT CYCLE
IN 13	FETCH CYCLE
IN 14	I/O OUTPUT CYCLE
IN 15	MEM READ CYCLE

OUT 15	SINGLE STEP	A logical '1' will cause the WAIT signal to be interrupted for 1 microsecond to allow the CPU to advance one machine cycle through the stored program.

LOAD ADDR mode

F	F	F	F	1	Ø	Ø	Ø	D	D	D	D	D	D	D	D

OUT Ø ----- OUT 7	ADDR	Either the low-order or high-order byte of the address depending on the value of OUT 15.
OUT 8	RESET	Logical 'Ø'
OUT 9	HOLD	Logical 'Ø'
OUT 1Ø	WAIT	Logical 'Ø'
OUT 11	LOAD ADDR	A logical '1' will cause the contents of OUT Ø to OUT 7 to be loaded into the interface address buffer according to the value of OUT 15.
OUT 12		Not used
OUT 13		Not used
OUT 14		Not used
OUT 15	$\overline{\text{LOW}}$/HIGH	A logical 'Ø' will cause the data in OUT Ø to OUT 7 to be interpreted as a low-order byte during a LOAD ADDR

operation and a logical '1' will cause it to be interpreted as a high-order byte.

RESET AND WAIT mode | F | F | F | F | Ø | 1 | Ø | 1 | D | D | D | D | D | D | D | D |

This mode of operation causes a full RESET of the microprocessor as previously described but execution of the program is suspended at the first instruction (memory location zero).

RUN AND LOAD ADDR mode | F | F | F | F | 1 | Ø | Ø | Ø | D | D | D | D | D | D | D | D |

The design of the interface circuitry allows loading of the interface address buffer while normal processor operation is continuing. This facility is useful to save DMA execution time.

2.3 THE EXTERNAL INTERFACE CARD - THEORY OF OPERATION

2.3.1 Direct Memory Access Section

In order to follow the operation of this, or any other, section of the Interface Card it is essential to have a thorough understanding of the I8Ø8Ø microprocessor system as explained in the reference manual[67].

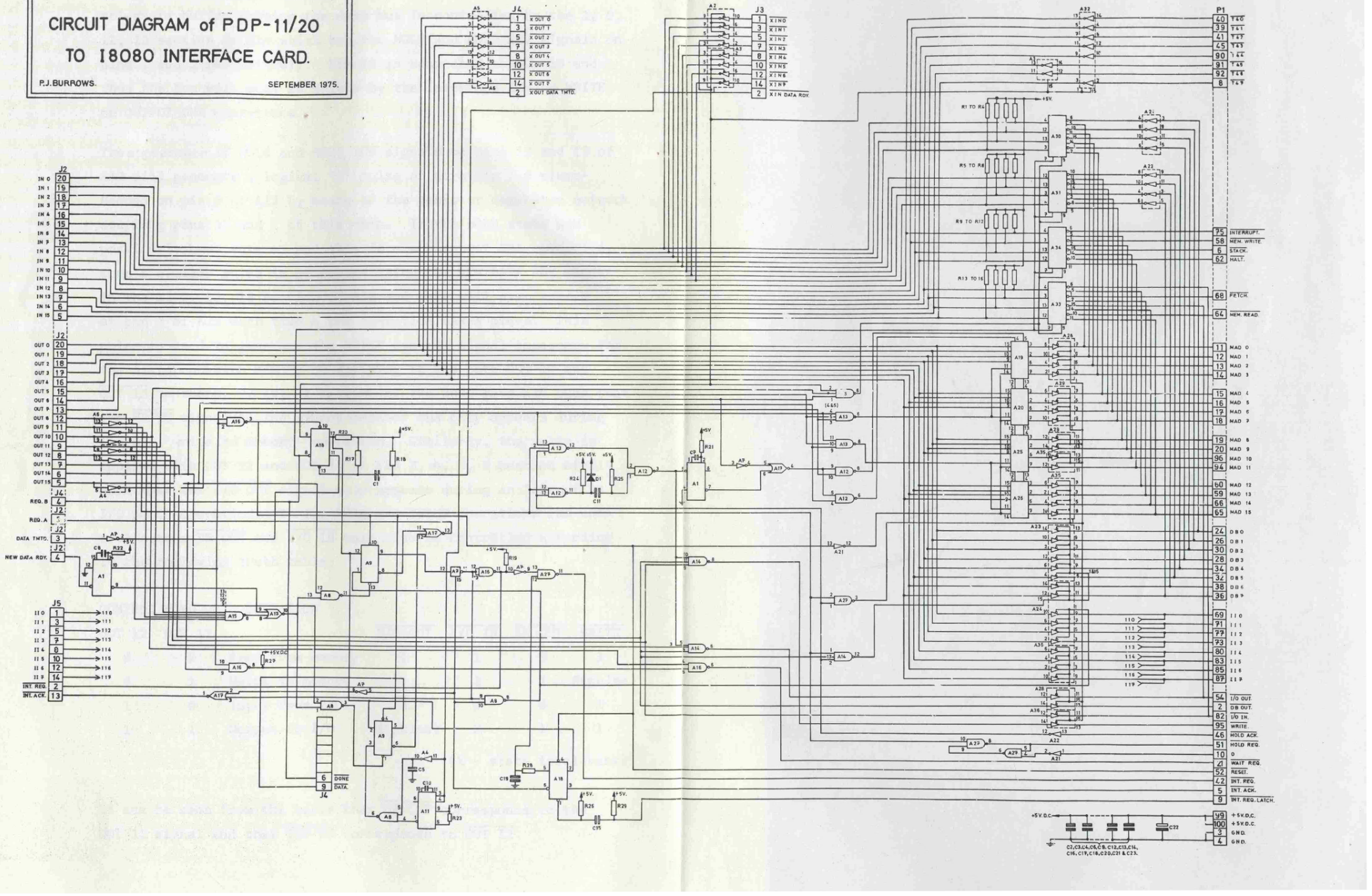
The DMA mode of operation is initiated by the OUT 9 signal line being driven to a logical '1' level by the PDP-11/2Ø or other external circuitry. This appears on pin 1Ø of A27 (refer to Fig 2.2). This gate has the processor ϕ_1 clock signal appearing on pin 9 via gates of A21 and A27. Thus the HOLD request is clocked to coincide correctly with the processor's internal timing and is presented to the processor as a $\overline{\text{HOLD REQ}}$ signal at the correct instant. When it is able to free the busses, the processor acknowledges with a $\overline{\text{HOLD ACK}}$ signal and this is inverted and used with the HOLD signal to prime certain gates of A12, A13, A14 and A27 as shown on Fig 2.2. The appearances of both these signals on pins 1 and 2 of the NAND gate A27 causes pin 3 to go 'low' thus enabling the tri-state buffers A28, A29, A35 and part

Fig. 2.2 PDP-11/20 to I8080 Interface Card Circuit Diagram

CIRCUIT DIAGRAM of PDP-11/20 TO 18080 INTERFACE CARD.

P.J. BURROWS.

SEPTEMBER 1975.



of A36 to take command of the address and control busses. The tri-state buffer driving the data bus is controlled by the 1, 2, 12, 13 section of A14 which has the HOLD and HOLD ACK signals on pins 1 and 2 respectively. Pin 13 is connected to OUT 13 and thus the bus will only be driven by the interface during WRITE or OUTPUT DMA operations.

The appearance of HOLD and HOLD ACK signals on pins 12 and 13 of A12 will generate a logical '1' pulse of duration 1.Ø microsecond on pin 3 of A12 by means of the resistor/capacitor network coupling pins 11 and 1 of this chip. If the HOLD state has previously been entered and this is a subsequent DMA operation, then a similar pulse is generated using the NEW DATA RDY signal in the 1, 2, 12, 13 section of A13 and this will also appear at pin 3 of A12 each time a DMA operation takes place. This pulse is used to generate the WRITE control signal necessary for the microprocessor memory by applying it and both $\overline{\text{OUT 12}}$ and $\overline{\text{OUT 13}}$ to the 8, 9, 1Ø, 11 section of the NAND gate A14 such that the $\overline{\text{WRITE}}$ signal on the microprocessor bus only appears during the DMA 'write to memory' operation. Similarly, the pulse is applied with OUT 12 and OUT 13 to the 3, 4, 5, 6 section of A14 such that the $\overline{\text{I/O OUT}}$ signal only appears during an 'output to I/O port' operation. During READ and INPUT operations the control busses $\overline{\text{DB OUT}}$ and $\overline{\text{I/O IN}}$ must also be controlled according to the following truth table:

CONTROL WORD		OPERATION				
OUT 12	OUT 13		$\overline{\text{I/O OUT}}$	$\overline{\text{I/O IN}}$	$\overline{\text{DB OUT}}$	$\overline{\text{WRITE}}$
Ø	Ø	Read from memory	1	1	Ø	1
Ø	1	Write to memory	1	1	1	Ø(pulse)
1	Ø	Input from I/O	1	Ø	Ø	1
1	1	Output to I/O	Ø(pulse)	X	1	1

(X - state irrelevant)

It can be seen from the table that $\overline{\text{DB OUT}}$ corresponds to the OUT 13 signal and that $\overline{\text{I/O IN}}$ corresponds to $\overline{\text{OUT 12}}$.

The trailing edge of the pulse at A12, pin 3 causes the monostable multivibrator A1 to trigger. If the 'auto-increment' mode is specified (OUT 14 = '1'), then pin 5 of A17 will be 'low' and the monostable pulse will be transmitted via this gate to pins 5 and 9 of A12. Depending on the state of OUT 15, either pin 4 or pin 1Ø will be 'high' and so the pulse will be transmitted to either pin 4 or 5 of A19, causing the address counter chain, comprising A19, A20, A25 and A26, to count up or down a single count. It is important to note that this occurs, in general, approximately 1.5 microseconds after the appearance of the control word from the PDP-11/2Ø such that, in the WRITE mode, the address counter points to the next (unwritten) address. However, in the READ mode the output of the control word by the PDP-11/2Ø must, of necessity, be followed by an input operation which may occur several microseconds later. Thus the incrementing of the address counter will appear to have occurred prior to the READ operation.

In both the READ and WRITE modes of operation, the multiplexers A3Ø, A31, A33 and A34 have pin 7 'low' and pin 9 'high', thus selecting the DATA and CPU STATUS signals to the outputs IN Ø to IN 15. As well as being essential for the READ operation, this facility also allows data written to be checked for transmission errors, if required, by means of the PDP-11/2Ø subsequently reading the data on IN Ø to IN 7 and comparing it with that transmitted.

2.3.2 Control and De-bugging Section

A RESET of both the microprocessor CPU and the interface is initiated by the appearance of a logical '1' signal level on OUT 8. This signal is clocked in A16 with a 1.5 microsecond pulse from the monostable multivibrator A1, which is triggered by the NEW DATA RDY signal from the PDP-11/2Ø. The output of A16 is applied to the microprocessor bus $\overline{\text{RESET}}$ line which restores the processor's internal program counter and instruction register to zero to restart a program. The output pulse from A16, pin 3, is also applied to the 'preset' inputs of both latches in each of A9 and A18 thus setting the 'Q' outputs (pins 5 and 9) to logical '1' and the '$\bar{Q}$' outputs (pins 6 and 8) to logical 'Ø'.

As will be seen later, this indicates to both processors that no data transfers or interrupts are in progress.

The $\overline{\text{RESET}}$ pulse is also inverted in the 12, 13 section of A21 and applied to the 'clear' control (pins 14) of the address counter, A19, A2Ø, A25 and A26, and thus this is also reset to zero by OUT 8.

To force a WAIT condition of the microprocessor requires the $\overline{\text{WAIT REQ}}$ bus being held in a logical 'Ø' state. This bus must be driven by an open-collector output as the various memory devices also control the bus under certain conditions. The signal is generated in the interface by the open-collector NAND gate A16 whose output is at pin 6. One of the inputs to this gate is directly derived from OUT 1Ø and the other is derived from another NAND gate (pin 8) in the same chip. Assuming that pin 8 is normally 'high', then a WAIT state will be caused at the next T_w state of the microprocessor CPU following OUT 1Ø going 'high'. OUT 15 and the NEW DATA RDY pulse from A1 are applied to pins 9 and 1Ø of A16 causing pin 8 to go 'low' for 1.5 microseconds whenever OUT 15 is 'high'. This signal is applied to pin 5 of A16, causing the $\overline{\text{WAIT REQ}}$ to be interrupted for sufficient time to allow the CPU to commence the next machine cycle and thus enabling the PDP-11/2Ø to control the SINGLE STEP operation of the microprocessor.

Normally, the control pins 7 and 9 of the multiplexers A3Ø, A31, A33 and A34 will be respectively 'Ø' and '1' such that the output word IN Ø to IN 15 displays the contents of the DATA bus and the CPU STATUS during a WAIT state. However, the NAND gate A15 (pins 1, 2, 4, 5 and 6) is driven by the signals $\overline{\text{REQ B}}$ (A18, pin 9), OUT 1Ø and OUT 14 such that in the WAIT state with no interrupt in progress (an interrupt is signified by REQ B being 'high') and OUT 14 being 'high', then the multiplexer control pin 9 goes 'low'. Furthermore, as the multiplexer control pin 7 is also driven by REQ B (A18, pin 8), this will also be 'low' under these conditions and thus the memory address bus, MAD Ø to MAD 15, will be switched to IN Ø to IN 15 and thus made available to the PDP-11/2Ø. It should be noted that an interrupt

occurring in the SINGLE STEP or WAIT mode will cause REQ B to go 'high' thus causing both pins 7 and 9 of the multiplexer to be logical '1', thereby disabling the multiplexer output. This is necessary as, under these conditions, the signals IN Ø to IN 15 are driven by the microprocessor output ports as explained in greater detail in the discussion of interrupt operation (Section 2.3.3).

2.3.3 The Interrupt Section

The interrupt section of the interface circuitry controls interrupts of either processor by the other, as well as the servicing of microprocessor peripheral interrupts. The operation is best explained by considering a typical interrupt and some of the considerations of the associated software. Firstly, consider an interrupt of the PDP-11/2Ø by the I8Ø8Ø microprocessor. In general, the microprocessor has the ability, by means of the interface, to transmit two 8-bit bytes as it generates the interrupt request and the significance of these bytes will depend on the particular programs in operation. Typically, however, these two bytes will represent a 16-bit data word as input to the PDP-11/2Ø.

If the microprocessor wishes to transmit both of the two 8-bit bytes, then the first (normally high-order) must be output to the Programmable Output Port (T4Ø to T47). This stage has no effect on the interface circuitry of the PDP-11/2Ø as the buffers in A32 and part of A3 are disabled by their controls (pins 1 and 15) being 'high'. If no data is being transferred via this port, then the microprocessor may proceed directly to the next operation, which is to load an 8-bit byte (normally low-order) into the 'XIN' port of the interface. This port is normally connected to Output Port Ø of the microprocessor system and so this loading can be performed by an 'OUT Ø' instruction. As this is done, the control signal (socket J4, pin 2) goes 'low' which causes, via the resistor/capacitor network, a temporary 'low' on pin 13 of A18. This causes the latch in that section of A18 to be cleared, such that its 'Q' output (pin 9) is logical 'Ø' and its '$\bar{Q}$' output (pin 8) is logical '1'.

The '$\bar{Q}$' output is applied to pin 7 of the multiplexer A3Ø, A31, A33 and A34 and the 'Q' output is applied to pin 2 of the NAND gate A15. The latter is such that, regardless of the other inputs to this gate, the output (pin 6) and thus pin 9 of the multiplexer will be 'high'. Thus the multiplexer is disabled. The 'Q' output is also connected to the tri-state buffers, A2, A3, A34, which are enabled at this time connecting the Programmable Display Port (T4Ø to T47) and Output Port Ø (XIN Ø to XIN 7) to the transfer bus IN Ø to IN 15 to be made available to the PDP-11/2Ø. The output '$\bar{Q}$', which has gone to logical '1', is connected to the REQ B input of the PDP-11/2Ø, such as to cause an interrupt if the DR11-A device has been previously 'interrupt enabled'. The same signal is also fed back into the microprocessor via J4, pin 6 and (normally) made available at Input Port 1, bit 7 to signify that an interrupt or transfer is in progress. The microprocessor software should then delay any further interrupts or data transfers until this bit indicates that the PDP-11/2Ø has read the data currently being transferred to it. (Note: care must be taken on account of the logical complementing of data that occurs on input to the microprocessor. If this bit is simply input and then tested it has the significance of a DONE bit:-

'Ø' - Interrupt in progress

'1' - Interrupt DONE, no interrupt in progress)

When the PDP-11/2Ø has read the data presented to it, it sends out a signal (DATA TMTD) which is applied to pin 11 of the latch restoring it to its original condition of 'Q' = '1' and '$\bar{Q}$' = 'Ø'. This indicates to the microprocessor, via the DONE bit, that further interrupts or transfers may follow.

If the PDP-11/2Ø wishes to interrupt the microprocessor, then it must supply a single-byte interrupt instruction to the Interrupt Instruction Port (II Ø to II 7). This is achieved in the following manner. If the interface is currently allowing the microprocessor to be in the RUN mode (OUT 8 to OUT 11 at logical 'Ø'), then pin 8 of the 4-input NAND gate A15 will be 'low'. Thus, only under this condition, will the 1.5 microsecond pulses from A1 be available at pin 1Ø of the NOR gate A17. If an interrupt is requested, then pin 12 of A8 will go 'high', which permits the

pulse to pass through the 11, 12, 13 section of this NAND gate to the 'clear' input of the latch in A9 (pin 13). The presence of this request is fed back to the PDP-11/2Ø via the 11, 12, 13 section of A17 to the REQ A input to act as a DONE bit. This DONE bit has the same significance as that mentioned immediately above.

At this stage in the flow of the INT REQ signal to the microprocessor, the Interface Card allows for the presence of other interrupt request signals from the microprocessor peripherals located in the hardware stack at socket J5. In the immediate discussion of the PDP-11/2Ø generated interrupt, it is assumed that no other such request signals are present. Thus, referring to Fig 2.3, the 'Q' output of the latch (A9, pin 9) will go 'low' to be inverted in the 11, 12 section of the latch A7 as its control (pin 15) is also 'low'. This latter signal is the '$\bar{Q}$' output (pin 6) of a latch in A18. This PDP-11 INT REQ signal is then applied to pins 12 and 13 of the open-collector NAND gate A16 causing its output, the $\overline{\text{INT REQ}}$ line common to all peripherals, to go 'low'. It is necessary that A16 is an open-collector device, because of the common access of all peripherals to this request line. This signal is then inverted and gated through the 11, 12, 13 section of A27 to be applied to the $\overline{\text{INT REQ}}$ line of the microprocessor. The microprocessor acknowledges immediately with the $\overline{\text{INT REQ LATCH}}$ signal which, by means of a resistor/capacitor network, clears the 'Q' output (pin 5) of the latch in A18.

This setting of the 'Q' output to logical 'Ø' has two effects. The first is to close the 11, 12, 13 section of gate A27 thus removing the $\overline{\text{INT REQ}}$ from the microprocessor and ensuring that, once this request has been serviced, any further request outstanding on the common peripheral $\overline{\text{INT REQ}}$ line will be re-applied to the microprocessor as a separate pulse generating another interrupt. The second action is by means of the '$\bar{Q}$' output (A18, pin 6) which goes 'high' and locks the 11, 12 section of latch A7 thus preventing the PDP-11/2Ø from generating an interrupt if the request had been initiated by one of the microprocessor peripherals. As the request in this case was

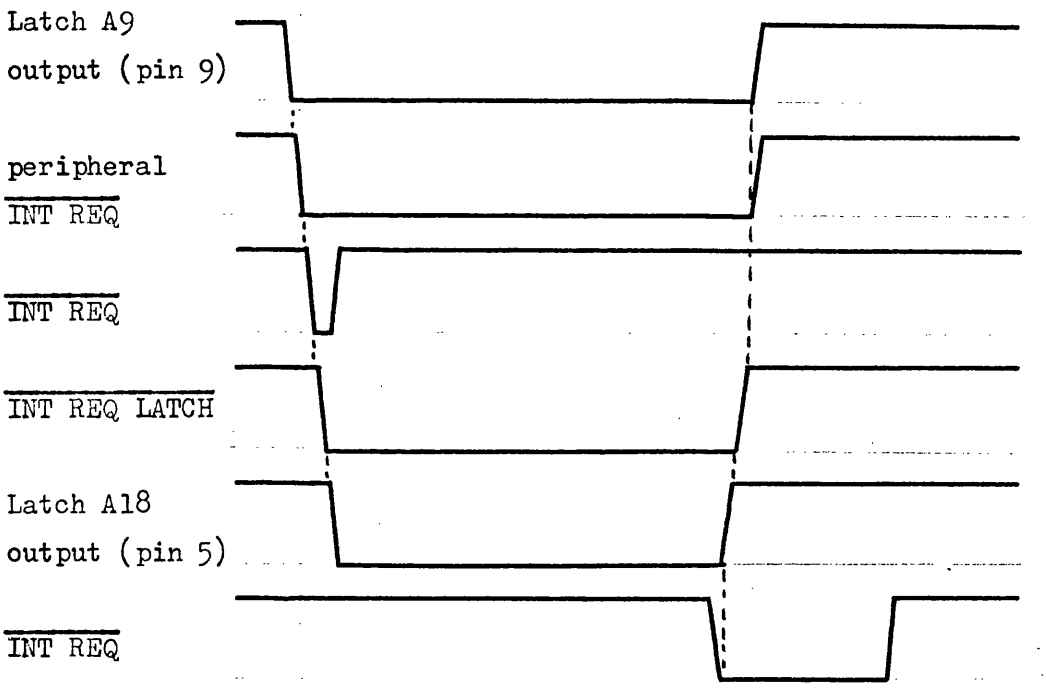


Fig 2.3 Timing Diagram for PDP-11/2Ø Interrupt Request

generated by the PDP-11/2Ø, then both pins 9 and 1Ø of the NAND gate A8 will be 'high' and its output 'low'. This logical 'Ø' is applied to pin 1 of each of the tri-state buffers A24 and A26 which are thus enabled and so apply bits OUT Ø to OUT 7 from the PDP-11/2Ø to the Interrupt Instruction Port (II Ø to II 7).

After completion of the current instruction, the microprocessor will enter an INTERRUPT CYCLE and will acknowledge the interrupt with an $\overline{\text{INT ACK}}$ signal and will interpret the interrupt instruction according to its type. The $\overline{\text{INT ACK}}$ signifies that the microprocessor has read the data on II Ø to II 7 as an instruction and it may now be removed. This signal is applied to pin 3 of A18 to reset this latch which, in turn, re-opens the 11, 12, 13 section of gate A27. There is a short delay in this re-opening to allow for the resetting of the latch in A9 via the 8, 9, 1Ø section of A8. Thus, ultimately, all latches are reset to their original condition and the DONE signal (REQ A) to the PDP-11/2Ø is issued.

The interrupt section of the interface also acts as the base

section of the hardware interrupt priority stack for the microprocessor peripherals (Section 6). Consider a typical peripheral interrupt stage as shown in Fig 2.4. An interrupt is initiated by the setting of the 7474 latch section which causes its '$\bar{Q}$' output to go 'low' thus, via the 34Ø4 latch and 74Ø5 open-collector gate, causing the $\overline{\text{INT REQ}}$ line to be pulled low. This signal appears at pin 2 of socket J5 on the Interface Card (Fig 2.2) causing a similar sequence of events to those described for PDP-11 interrupts. The difference is, however, that now the 'high' on pin 6 of A18 will latch the 11, 12 section of A7 and prevent the PDP-11/2Ø from interrupting. Thus pin 11 of A7 will be held low and pins 2 and 3 of A17 will both be low, causing an INT ACK signal to be sent off the Interface Card via J5, pin 13. Referring again to Fig 2.4, and assuming that the first stage (highest priority) was not the one requesting the interrupt, then the output of the 34Ø4 gate will be low and the INT ACK signal will be transmitted past this stage via the 74Ø4 and 74Ø2 gates. Note that the INT ACK signal also serves to lock the 34Ø4 latch to prevent any interrupt from this stage having any effect once the acknowledgement has been passed to a lower priority stage.

When the INT ACK signal reaches the first stage which has requested an interrupt (there may have been more than one occuring simultaneously), the 74ØØ gate is opened and the 74125 tri-state buffers are enabled, thus putting the hard-wired interrupt instruction onto the Interrupt Instruction Bus, IIØ to II7. The 7474 latch is also reset at this point in time. The 'high' output of the 34Ø4 has also closed the 74Ø2 gate, thus preventing the INT ACK signal reaching any further stages. After the reading of the interrupt instruction by the microprocessor, pin 6 of latch A18 on the Interface Card is again restored to logical 'Ø' as previously and, via the 1, 2, 3 section of A17, the INT ACK signal is removed. After a short delay, the 11, 12, 13 section of A27 is again re-opened to test for any remaining requests on the common peripheral $\overline{\text{INT REQ}}$ line.

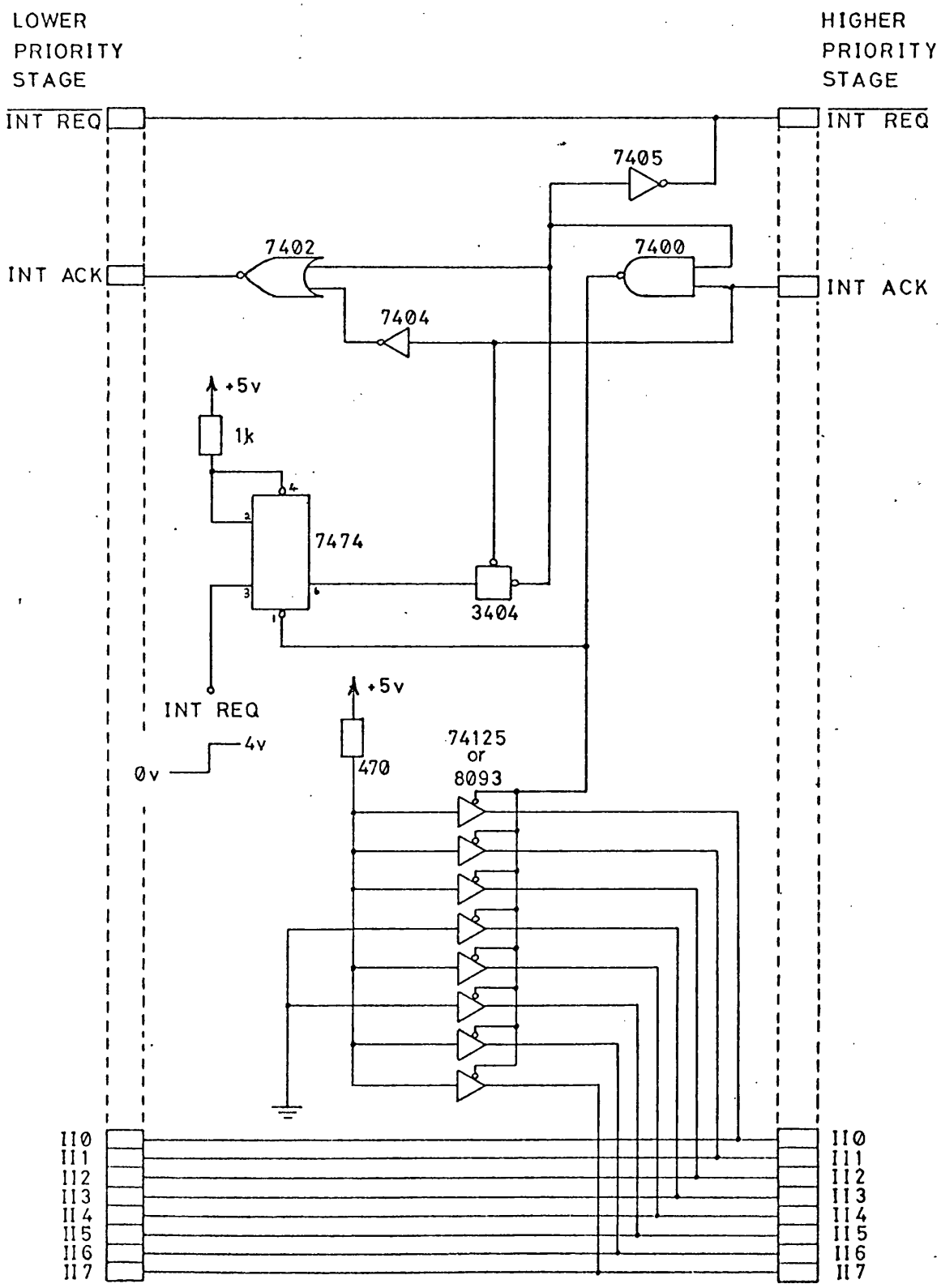


Fig. 2.4 Typical Interrupt Stage Schematic

2.3.4 The Data Transfer Section

Data transfers between the I8Ø8Ø microprocessor and the PDP-11/2Ø computer can be performed by means of an interrupt as previously described. If it is desired to transfer data under non-interrupt program control, then it is only necessary to disable the interrupt ability of the DR11-A interface and perform all data transfers under program control using the REQ B signal as a data transfer flag. (Note: In this mode the REQ B signal will have the significance of a $\overline{\text{DONE}}$ bit, being 'low' when the data transfer is completed.

Data transfers to the I8Ø8Ø without interrupt are enabled in the RUN mode by the presence of a logical '1' at OUT 14. As described during the discussion of interrupt operation, pin 1Ø of A17 will only have a NEW DATA RDY pulse during RUN mode operations and this is gated by OUT 14 in the 1, 2, 3 section of A8 to pin 1 of A9. This latch then sends a logical 'Ø' from pin 5 to socket J4, pin 9. This is normally connected to Input Port 1, bit 6, and when read (and complemented by the input operation), will appear within the microprocessor as a logical '1', signifying DATA ready. This latch also indicates, via the 11, 12, 13 section of A17 that a data transfer is in progress, the REQ A acting as a DONE bit, being 'low' while the data transfer is in progress. The data on OUT Ø to OUT 7 appears at XOUT Ø to XOUT 7, having passed via the inverters in A5 and A6 to compensate for the complementing of data which occurs in a microprocessor input operation. The XOUT port (J4) is normally connected to Input Port Ø of the microprocessor and thus made available to it. During an INput operation, a 'low' pulse will appear on J4, pin 2. This pulse is of duration less than 1.Ø microseconds if the INput operation is from a port other than Port Ø and of approximately 1.5 microseconds if the operation is from Port Ø. The duration of the pulse is timed by the monostable A11 and its associated gates and if an 'IN Ø' operation has occured the latch in A9 is reset, thus indicating the end of the data transfer.

2.4 THE EXTERNAL INTERFACE CARD - UTILIZATION

2.4.1 Input and Output Connections

All logic inputs and outputs to the External Interface Card are compatible with conventional TTL logic levels, with the exception of those outputs to the microprocessor bus which use open-collector and tri-state logic for bus compatibility, as described in previous sections.

The connector details are given below.

SOCKET J1 - Interface Output

Pin	Name	Signal Function
1		Not Used
2	GND	Signal Ground
3	REQ A	Data Transfer 'DONE'
4	REQ B	Interrupt Request
5	IN 15	Output Bit 15
6	IN 14	" " 14
7	IN 13	" " 13
8	IN 12	" " 12
9	IN 11	" " 11
10	IN 10	" " 10
11	IN 9	" " 9
12	IN 8	" " 8
13	IN 7	" " 7
14	IN 6	" " 6
15	IN 5	" " 5
16	IN 4	" " 4
17	IN 3	" " 3
18	IN 2	" " 2
19	IN 1	" " 1
20	IN 0	" " 0

Note: Socket J1 will normally be directly or indirectly connected to the DR11-A input of the PDP-11/20 computer but may alternatively be connected to display lights or other compatible system.

SOCKET J2 - Interface Input

Pin	Name	Signal Function
1		Not Used
2	GND	Signal Ground
3	DATA TMTD	Data Transmitted
4	NEW DATA RDY	New Data Ready
5	OUT 15	Input Bit 15
6	OUT 14	" " 14
7	OUT 13	" " 13
8	OUT 12	" " 12
9	OUT 11	" " 11
1Ø	OUT 1Ø	" " 1Ø
11	OUT 9	" " 9
12	OUT 8	" " 8
13	OUT 7	" " 7
14	OUT 6	" " 6
15	OUT 5	" " 5
16	OUT 4	" " 4
17	OUT 3	" " 3
18	OUT 2	" " 2
19	OUT 1	" " 1
2Ø	OUT Ø	" " Ø

Note: Socket J2 will normally be directly or indirectly connected to the DR11-A output of the PDP-11/2Ø computer but may alternatively be connected to suitably buffered switches or other compatible system.

SOCKET J3 - Data Input Port

Pin	NAME	Signal Function	Connect To
1	$\overline{\text{XIN Ø}}$	Data Bit Ø	Output Ø, Bit Ø
2	$\overline{\text{XIN DATA RDY}}$	" Ready	Output Ø, Data Ready
3	$\overline{\text{XIN 1}}$	" Bit 1	Output Ø, Bit 1
4		Not Used	
5	$\overline{\text{XIN 2}}$	Data Bit 2	Output Ø, Bit 2
6		Not Used	
7	$\overline{\text{XIN 3}}$	Data Bit 3	Output Ø, Bit 3
8	$\overline{\text{XIN 4}}$	" " 4	Output Ø, Bit 4
9		Not Used	
1Ø	$\overline{\text{XIN 5}}$	Data Bit 5	Output Ø, Bit 5
11		Not Used	
12	$\overline{\text{XIN 6}}$	Data Bit 6	Output Ø, Bit 6
13		Not Used	
14	$\overline{\text{XIN 7}}$	Data Bit 7	Output Ø, Bit 7

Note: Socket J3 will normally be directly connected to the micro-

processor system Output Port Ø but may be connected to any other output port if software considerations so permit.

SOCKET J4 - Data Output Port

Pin	Name	Signal Function	Connect To
1	$\overline{\text{X OUT Ø}}$	Data Bit Ø	Input Ø, Bit Ø
2	$\overline{\text{XOUT DATA TMTD}}$	" Transmitted	Input Ø, Data Tmtd
3	$\overline{\text{XOUT 1}}$	Data Bit 1	Input Ø, Bit 1
4		Not Used	
5	$\overline{\text{XOUT 2}}$	Data Bit 2	Input Ø, Bit 2
6	$\overline{\text{DONE}}$	" Transmission Done	Input 1, Bit 7
7	$\overline{\text{XOUT 3}}$	" Bit 3	Input Ø, Bit 3
8	$\overline{\text{XOUT 4}}$	" " 4	Input Ø, Bit 4
9	$\overline{\text{DATA RDY}}$	" Ready	Input 1, Bit 6
1Ø	$\overline{\text{XOUT 5}}$	" Bit 5	Input Ø, Bit 5
11		Not Used	
12	$\overline{\text{XOUT 6}}$	Data Bit 6	Input Ø, Bit 6
13		Not Used	
14	$\overline{\text{XOUT 7}}$	Data Bit 7	Input Ø, Bit 7

Note: Socket J4 will normally be directly connected to the microprocessor system Input Ports Ø and 1 but may be connected to any other input ports if software considerations so permit.

SOCKET J5 - Interrupt Port

Pin	Name	Signal Function
1	$\overline{\text{II Ø}}$	Interrupt Instruction Bit Ø
2	$\overline{\text{INT REQ}}$	" Request
3	II 1	" Instruction Bit 1
4	GND	Signal Ground
5	II 2	Interrupt Instruction Bit 2
6		Not Used
7	II 3	Interrupt Instruction Bit 3
8	II 4	" " " 4
9		Not Used
1Ø	II 5	Interrupt Instruction Bit 5
11		Not Used
12	II 6	Interrupt Instruction Bit 6
13	INT ACK	" Acknowledge
14	II 7	" Instruction Bit 7

EDGE CONNECTOR P1 - Microprocessor Bus Connector

Pin	Name	Signal Function
1		Not Used
2	DB OUT	Output Data Enabling
3	GND	Supply Common
4	GND	Supply Common
5	INT ACK	Interrupt Cycle Status
6	STACK	Stack Reference Cycle Status
7		Not Used
8	T47	Programmed Display Bit 7
9	INT REQ LATCH	Interrupt Requested
10	ø1	ø1 Processor Clock Out
11	MADØ	Address Bit Ø
12	MAD1	" " 1
13	MAD2	" " 2
14	MAD3	" " 3
15	MAD4	" " 4
16	MAD5	" " 5
17	MAD6	" " 6
18	MAD7	" " 7
19	MAD8	" " 8
20	MAD9	" " 9
21	WAIT REQ	Ready Flag From Memory
22		Not Used
23		" "
24	DBØ	Output Data Bit Ø
25		Not Used
26	DB1	Output Data Bit 1
27		Not Used
28	DB3	Output Data Bit 3
29		Not Used
30	DB2	Output Data Bit 2
31		Not Used
32	DB5	Output Data Bit 5
33		Not Used
34	DB4	Output Data Bit 4
35		Not Used
36	DB7	Output Data Bit 7
37		Not Used
38	DB6	Output Data Bit 6
39	T41	Programmed Display Bit 1
40	T4Ø	" " " Ø
41	T42	" " " 2
42	INT REQ	Initiate External Interrupt
43		Not Used
44		" "
45	T43	Programmed Display Bit 3
46	HOLD ACK	Acknowledge Hold Request
47		Not Used
48		" "
49		" "
50		" "

Pin	Name	Signal Function
51	HOLD REQ	Initiate External Hold
52	RESET	" " Reset
53		Not Used
54	I/O OUT	I/O Output Strobe
55		Not Used
56		" "
57		" "
58	MEM WRITE	Memory Write Cycle Status
59	MAD13	Address Bit 13
6Ø	MAD12	" " 12
61		Not Used
62	HALT	Halt Cycle Status
63		Not Used
64		" "
65	MAD15	Address Bit 15
66	MAD14	" " 14
67	MEM READ	Memory Read Cycle Status
68	FETCH	Instruction Fetch Cycle Status
69	IIØ	Interrupt Instruction Bit Ø
7Ø		Not Used
71	II1	Interrupt Instruction Bit 1
72		Not Used
73	II3	Interrupt Instruction Bit 3
74		Not Used
75	INT CYCLE	Interrupt Cycle Status
76		Not Used
77	II2	Interrupt Instruction Bit 2
78		Not Used
79		" "
8Ø	II4	Interrupt Instruction Bit 4
81		Not Used
82	I/O IN	I/O Input Strobe
83	II5	Interrupt Instruction Bit 5
84		Not Used
85	II6	Interrupt Instruction Bit 6
86		Not Used
87	II7	Interrupt Instruction Bit 7
88		Not Used
89		" "
9Ø	T44	Programmed Display Bit 4
91	T45	" " " 5
92	T46	" " " 6
93		Not Used
94	MAD11	Address Bit 11
95	WRITE	Memory Write Strobe
96	MAD1Ø	Address Bit 1Ø
97		Not Used
98		" "
99	+5 VDC	V_{cc} Source Power
1ØØ	+5 VDC	" " "

2.4.2 Installation Data

Connector to microprocessor bus:-

Dual 5Ø-pin PC Edge Connector. Ø.125" centres.
(Compatible with SAE type C8ØØ1ØØ
or CDC type VPBØ1C5ØEØØA1
or Viking Industries 3VH5Ø/1CN5)

Connector to microprocessor input/output and interrupt ports:-

14-pin Dual-in-line socket. Ø.1" centres.
(Compatible with Jermyn type A23-2Ø48LC)

Connector to Line Driver/Receiver Card:-

2Ø-pin Single-in-line socket. Ø.1" centres.
(Compatible with Jermyn type A23-2Ø74PS)

Power Supply Requirements:-

+5 V. dc ±5% @ 1.3 A typ. (2.Ø A. max)

Operating Temperature Range:-

+Ø°C to +7Ø°C

2.4.3 Component Specifications

Integrated Circuits

A1	=	96Ø2PC	A12	=	SN74ØØN
A2	=	SN74368N	A13	=	SN741ØN
A3	=	SN74368N	A14	=	SN741ØN
A4	=	SN74Ø4N	A15	=	SN742ØN
A5	=	SN74Ø4N	A16	=	SN74Ø3N
A6	=	SN74Ø4N	A17	=	SN74Ø2N
A7	=	P34Ø4	A18	=	SN7474N
A8	=	SN74ØØN	A19	=	SN74193N
A9	=	SN7474N	A2Ø	=	SN74193N
A11	=	SN7412N	A21	=	SN74Ø4N

Integrated Circuits cont

A22	=	SN74Ø4N	A3Ø	=	N8267B
A23	=	SN74367N	A31	=	N8267B
A24	=	SN74367N	A32	=	SN74368N
A25	=	SN74193N	A33	=	N8267B
A26	=	SN74193N	A34	=	N8267B
A27	=	SN74ØØN	A35	=	SN74367B
A28	=	SN74367N	A36	=	SN74367B
A29	=	SN74367N			

Diodes

D1	AA144
D2	AA144

Resistors

R1 to R2Ø	1 KΩ, $\frac{1}{8}$ Watt, metal oxide
R21	1Ø KΩ, " " "
R22	12 KΩ, " " "
R23	1 KΩ, " " "
R24	1 KΩ, " " "
R25	47Ø Ω , " " "
R26 to R29	1 KΩ, " " "

Capacitors

C1	15ØØ pf
C2 to C4	.Ø1 µf ceramic
C5	22ØØ pf
C6	.Ø1 µf ceramic
C7 to C8	33Ø pf
C9	.Ø1 µf ceramic
C1Ø	68Ø pf
C11	47ØØ pf
C12 to C14	.Ø1 µf ceramic
C15	15ØØ pf
C16 to C18	.Ø1 µf ceramic
C19	15ØØ pf
C2Ø to C21	.Ø1 µf ceramic
C22	8Ø µf, 25V electrolytic
C23	.Ø1 µf ceramic

Sockets

J1	A23-2Ø74PS
J2	A23-2Ø74PS
J3	A23-2Ø48LC or A23-2Ø28Z
J4	A23-2Ø48LC or A23-2Ø28Z
J5	A23-2Ø48LC or A23-2Ø28Z

Headers

J24-2148 (3 off)

2.4.4 Component Layout

The component layout is given in Fig 2.5

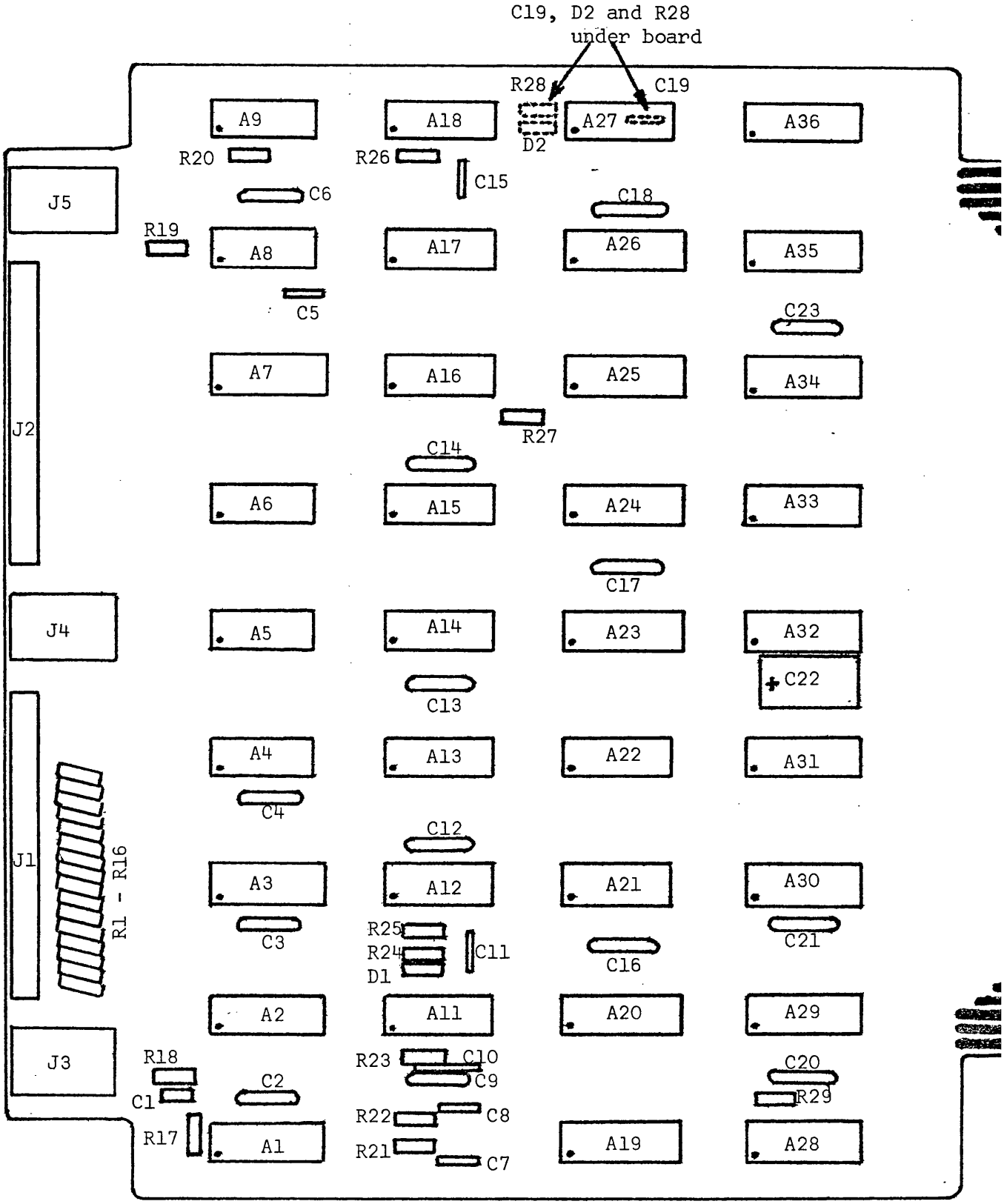


Fig 2.5 PDP-11/2Ø to I8Ø8Ø Interface Card Component Layout

(view from above)

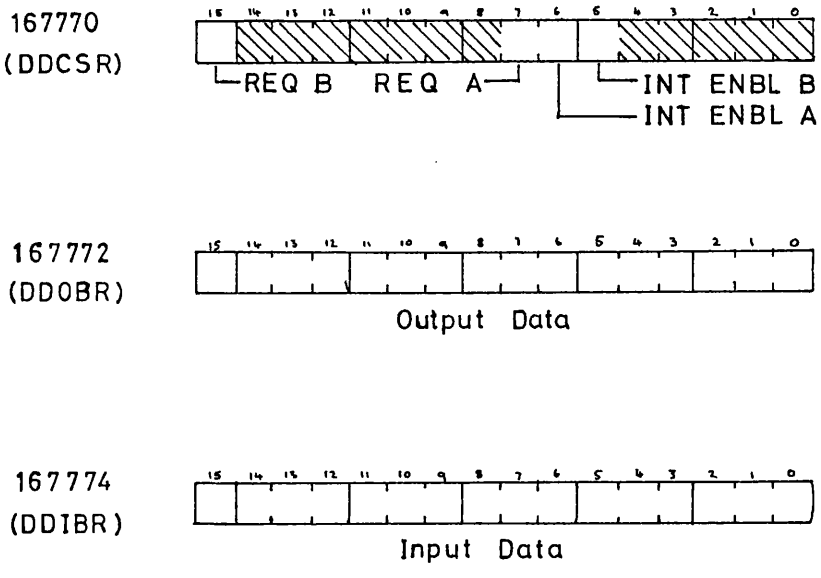
SECTION 3

GENERAL DEVICE INTERFACE

The PDP-11/20 computer communicates with the I8080 microprocessor via a DR11-A or DR11-C General Device Interface mounted within the main PDP-11 console. This unit provides a general purpose interface facility between the PDP-11 UNIBUS and a user's peripheral, in this case the microprocessor. A complete functional description of the DR11-C unit is given in the relevant device manual[68] and a summary of the DR11-A utilisation is given in the peripherals handbook[69].

The two units are similar in operation and software written for use with the earlier DR11-A is fully compatible with the DR11-C (once the necessary device address changes have been made (see Section 13)). The inverse, however, is not necessarily true as the later unit has an increased number of functions. The following description relates primarily to the DR11-A Interface which is in use for the communications link to the first micro-machine system.

3.1 DEVICE REGISTERS



3.1.1 Control and Status Register (DDCSR) 16777Ø

The control and status register has four bits of interest to the programmer. The two REQ (request) bits are under control of the PDP-11 to I8Ø8Ø Interface Card to provide status information and may be used to generate interrupts when used with their corresponding INT ENBL (interrupt enable) bits which are under program control. The bit functions are as follows:-

Bit	Name	Function
15	REQ B	DATA ready from I8Ø8Ø. This bit is set when data is output to microprocessor port Ø. Data may have been previously output to port 377_8 (see Section 2.3.3). This data will be available at the input register (DDIBR). This bit will cause an interrupt if INT ENBL B (Bit 5) is set by the program (or device handler). Read-only bit. This bit is cleared by a RESET (Bit 8 in DDOBR, see Section 2.2.5) or by a read operation of the DDIBR.
7	REQ A	DONE or READY on data transfer or interrupt of I8Ø8Ø by the PDP-11. This bit is cleared when the word $Ø4ØXXX_8$ (data) or $1ØØXXX_8$ (interrupt) is output to the DDOBR (see Sections 2.2 and 2.3) This bit will cause an interrupt if INT ENBL A (Bit 6) is set by the program (or device handler). Read-only bit. This bit is set by a RESET or by a microprocessor input operation of port Ø or by the microprocessor entering an interrupt cycle caused by the PDP-11.

6 INT ENBL A Interrupt enable bit. When set, allows an interrupt sequence to be initiated whenever REQ A is set, i.e. when a data transfer to the I8Ø8Ø is complete or when the I8Ø8Ø has acknowledged an interrupt by the PDP-11.

Read or Write bit. Cleared by INIT.

5 INT ENBL B Interrupt Enable Bit. When set, allows an interrupt sequence to be initiated whenever REQ B is set, i.e. whenever data is received from the I8Ø8Ø.

Read or Write bit. Cleared by INIT.

3.1.2 Output Buffer Register (DDOBR) 167772

The output buffer is a 16-bit read/write register that may be read or loaded under program control. The output is connected via the transmission link (section 4) to the PDP-11 to I8Ø8Ø Interface Card such that the bits have the special functions as described in Section 2.2. Loading certain control patterns signifying interrupts or data transfers will cause REQ A to be cleared.

3.1.3 Input Buffer Register (DDIBR) 167774

The input buffer is a 16-bit read-only register that receives information from the Interface Card. The significance of the information is dependent on the control word in the output buffer and the operation of the microprocessor or as described in Sections 2.2 and 2.3. However, the following brief summaries may be made:

1. If the microprocessor has output data to Port Ø (and possibly also Port 377_8) then the DDIBR will show the contents of those two ports and REQ B will be set. This has priority over 2 and 3 below.

2. If the control program has set a WAIT and DISP ADDR mode

(Bits 1Ø and 14 in the DDOBR) then the DDIBR will show the contents of the 16-bit microprocessor address bus.

3. In other modes the DDIBR will display the contents of the microprocessor data bus and status register as detailed in Section 2.2.

Any instruction which accesses the DDIBR will cause a DATA TMTD signal to be sent to the Interface which will then relinquish its hold of the buffer for data transfers and also clear REQ B. Thus great care should be exercised if the DDIBR is being examined while data transfers are in progress (Note: The debugging program DEBUG, Section 15, has a special facility to overcome this problem. Also if the standard device handlers are used no conflict can occur).

3.2 UTILIZATION

3.2.1 External Connections

The user connections to the DR11-A and DR11-C units are made available on the sockets of a card frame within the PDP-11 structure. The sockets used in the card frame are Vero type 13597/2 and accept cards of size 165.1mm x 114.3mm. Compatible card types are Vero type VB/10723/2 (vero-card) or 11822 (D.I.P. board).

Each signal line is available on two sockets, the pin connections being as follows and full explanation of the signal functions being given in the relevant manuals:-

Frame Positions 2 & 4

DR11-A Input (Connector No 2)

Connector Type M927A		Type 3365	Edge Connector Type 13597/2	
Pin No	Signal	Cable No	Pin No	Signal
			1*	ØVDC †
S1	INØØ	2	2	INØØ
	GND	1	3*	GND
	GND	3		
S2	INØ1	4	4	INØ1
	GND	5	5*	GND
P1	INØ2	6	6	INØ2
	GND	7	7*	GND
L1	INØ3	8	8	INØ3
	GND	9	9*	GND
P2	INØ4	10	10	INØ4
	GND	11	11*	GND
K2	INØ5	12	12	INØ5
	GND	13	13*	GND
M1	INØ6	14	14	INØ6
	GND	15	15*	GND
T2	INØ7	16	16	INØ7
	GND	17	17*	GND
M2	INØ8	18	18	INØ8
	GND	19	19*	GND
D2	INØ9	20	20	INØ9
	GND	21	21*	GND
E1	IN1Ø	22	22	IN1Ø
	GND	23	23*	GND
D1	IN11	24	24	IN11
	GND	25	25*	GND
H1	IN12	26	26	IN12
	GND	27	27*	GND
E2	IN13	28	28	IN13
	GND	29	29*	GND
	GND	31		
B1	IN14	30	30	IN14
	GND	33	31*	GND
	GND	35		
J1	IN15	32	32	IN15
			33	AUX A
	GND	34	34	GND
			35	AUX B
H2	REQ B	36	36	REQ B
	GND	37	37	GND
V2	DATA TMTD	38	38	DATA TMTD
	GND	39	39	GND
	GND	40	40	GND
			41*	ØVDC †
			42*	-5VDC
			43*	+5VDC

Frame Positions 6 & 8			DR11-A Output (Connector No 1)	
Connector Type M927A		Type 3365	Edge Connector Type 13597/2	
Pin No	Signal	Cable No	Pin No	Signal
			1*	ØVDC †
P2	OUTØØ	2	2	OUTØØ
	GND	1	3*	GND
	GND	3		
M2	OUTØ1	4	4	OUTØ1
	GND	5	5*	GND
S1	OUTØ2	6	6	OUTØ2
	GND	7	7*	GND
P1	OUTØ3	8	8	OUTØ3
	GND	9	9	GND
K2	OUTØ4	10	10	OUTØ4
	GND	11	11*	GND
M1	OUTØ5	12	12	OUTØ5
	GND	13	13*	GND
S2	OUTØ6	14	14	OUTØ6
	GND	15	15*	GND
L1	OUTØ7	16	16	OUTØ7
	GND	17	17*	GND
J1	OUTØ8	18	18	OUTØ8
	GND	19	19*	GND
H2	OUTØ9	20	20	OUTØ9
	GND	21	21*	GND
E2	OUT1Ø	22	22	OUT1Ø
	GND	23	23*	GND
H1	OUT11	24	24	OUT11
	GND	25	25*	GND
D2	OUT12	26	26	OUT12
	GND	27	27*	GND
E1	OUT13	28	28	OUT13
	GND	29	29*	GND
	GND	31		
D1	OUT14	30	30	OUT14
	GND	33	31*	GND
	GND	35		
B1	OUT15	32	32	OUT15
			33	AUX A
T2	REQ A	34	34	REQ A
			35	AUX B
	GND	36	36	GND
	GND	37	37	GND
V2	NEW DATA RDY	38	38	NEW DATA RDY
	GND	39	39	GND
	GND	40	40	GND
			41*	ØVDC †
			42*	-5VDC
			43*	+5VDC

Frame Positions 10 & 12

DR11-C Input (Connector No 2)

Berg Connector on M7860 Pin No	Signal	Type 3365 Cable No	Edge Connector Type 13597/2 Pin No	Signal
			1*	ØVDC †
TT	INØØ	38	2	INØØ
VV	OPEN	40	3*	GND
UU	GND	39		
LL	INØ1	32	4	INØ1
SS	GND	37	5*	GND
H	INØ2	7	6	INØ2
PP	GND	35	7*	GND
BB	INØ3	24	8	INØ3
MM	GND	33	9*	GND
KK	INØ4	31	10	INØ4
JJ	GND	30	11*	GND
HH	INØ5	29	12	INØ5
FF	OPEN	28	13*	GND
EE	INØ6	27	14	INØ6
DD	GND	26	15*	GND
CC	INØ7	25	16	INØ7
AA	GND	23	17*	GND
Z	INØ8	22	18	INØ8
X	GND	20	19*	GND
Y	INØ9	21	20	INØ9
T	GND	16	21*	GND
W	IN1Ø	19	22	IN1Ø
R	GND	14	23*	GND
V	IN11	18	24	IN11
L	GND	10	25*	GND
U	IN12	17	26	IN12
J	GND	8	27*	GND
P	IN13	13	28	IN13
F	OPEN	6	29*	GND
E	OPEN	5		
N	IN14	12	30	IN14
D	OPEN	4	31*	GND
B	OPEN	2		
M	IN15	11	32	IN15
			33	AUX C ‡
K	CSRØ	9	34	CSRØ
			35	AUX D ‡
S	REQ B	15	36	REQ B
A	OPEN	1	37	GND
C	DATA TMTD	3	38	DATA TMTD
NN	INIT H	34	39	INIT
RR	INIT H	36	40	INIT
			41*	ØVDC †
			42*	-5VDC
			43*	+5VDC

Frame Positions 14 & 16

DR11-C Output (Connector No 1)

Berg Connector on M7860 Type 3365			Edge Connector Type 13597/2	
Pin No	Signal	Cable No	Pin No	Signal
			1*	ØVDC †
C	OUTØØ	3	2	OUTØØ
UU	GND	39	3*	GND
TT	OPEN	38		
K	OUTØ1	9	4	OUTØ1
SS	GND	37	5*	GND
NN	OUTØ2	34	6	OUTØ2
RR	OPEN	36	7*	GND
U	OUTØ3	17	8	OUTØ3
PP	GND	35	9*	GND
L	OUTØ4	10	10	OUTØ4
MM	GND	33	11*	GND
N	OUTØ5	12	12	OUTØ5
KK	GND	31	13*	GND
R	OUTØ6	14	14	OUTØ6
EE	GND	27	15*	GND
T	OUTØ7	16	16	OUTØ7
CC	GND	25	17*	GND
W	OUTØ8	19	18	OUTØ8
Y	GND	21	19*	GND
X	OUTØ9	20	20	OUTØ9
V	GND	18	21*	GND
Z	OUT1Ø	22	22	OUT1Ø
S	GND	15	23*	GND
AA	OUT11	23	24	OUT11
M	GND	11	25*	GND
BB	OUT12	24	26	OUT12
J	GND	8	27*	GND
FF	OUT13	28	28	OUT13
F	OPEN	6	29*	GND
D	OPEN	4		
HH	OUT14	29	30	OUT14
B	OPEN	2	31*	GND
A	OPEN	1		
JJ	OUT15	30	32	OUT15
			33	AUX C ‡
LL	REQ A	32	34	REQ A
			35	AUX D ‡
DD	CSR1	26	36	CSR1
E	NEW DATA RDY HI	5	37	NEW DATA RDY HI
VV	NEW DATA RDY	40	38	NEW DATA RDY
H	NEW DATA RDY LO	7	39	NEW DATA RDY LO
P	INIT H	13	40	INIT H
			41*	ØVDC †
			42*	-5VDC
			43*	+5VDC

Notes:-

* Pins thus marked are parallel-connected across all frame positions. The GND signal is the PDP-11/2Ø signal ground from the DR11-A and DR11-C units.

† The ØVDC line is the zero voltage line from the power supply. If the +5V or -5V supply is being used,this line should be connected to one or more of the GND lines on the board being inserted.

‡ The auxilary lines AUX A to AUX D may be used to transmit control signals between the input and output cards of each DR11 unit. The connections are summarised as follows:-

Signal	Board Positions	Pin No
AUX A	2, 4, 6 and 8	33
AUX B	2, 4, 6 and 8	35
AUX C	10, 12, 14 and 16	33
AUX D	10, 12, 14 and 16	35

3.2.2 Installation Data

The DR11-A and DR11-C units are part of the main PDP-11 system and are powered from the PDP-11 internal power supplies. The card frame used for interfacing has two power supply rails nominally used to supply +5VDC and -5VDC to any cards inserted in the frame. These rails must be supplied with power from an external source either via one of the cards inserted or via the 4mm sockets on the card frame face. These supply rails are floating with respect to the DR11 outputs and should be ground referenced by connecting the supply ØVDC line (pins 1 or 41) to one of the DR11 GND lines (pins 3, 5, 7 etc) on the inserted card, should the application demand this.

SECTION 4

LINE DRIVER/RECEIVER CARD AND TRANSMISSION LINK

The PDP-11/2Ø computer is in a location remote from the microprocessor and micromachine model requiring a digital data transmission link approximately 4Ø feet in length. For simplicity and speed of commission, this link was constructed as two unidirectional 2Ø-bit data highways (allowing 16 data bits and 4 auxilary controls in each direction). The line length and impedance prohibited the use of standard TTL driver circuits and noise problems were experienced with single-ended buffer drivers and receivers.

4.1 FUNCTIONAL DESCRIPTION

The Line Driver/Receiver Card has 2Ø line driver circuits and 2Ø receivers and is duplicated at either end of the link thus providing the necessary 16-bit (plus auxilaries) communication in each direction between the PDP-11/2Ø and the I8Ø8Ø. The drivers and receivers function in a differential current-loop mode thus eliminating ground reference problems and severely limiting noise pick-up.

4.2 CIRCUIT DESCRIPTION

A circuit diagram of one channel of the transmission link is given in Fig 4.1. It can be seen that the line driver is a type 7511Ø which drives a current of 6 to 12 mA into the junction of the 75 Ω line and its terminating resistors. The line is symmetrically terminated about ground to reduce pickup. At the receiving end, the line is again symmetrically terminated about ground and the signal detected by the differential input of the 751Ø7 line receiver. This configuration shows a high common-mode rejection ratio (to reduce noise) and good sensitivity to transmitted signals. The configuration also allows for a small difference in earth potential between the two ends of the transmission link.

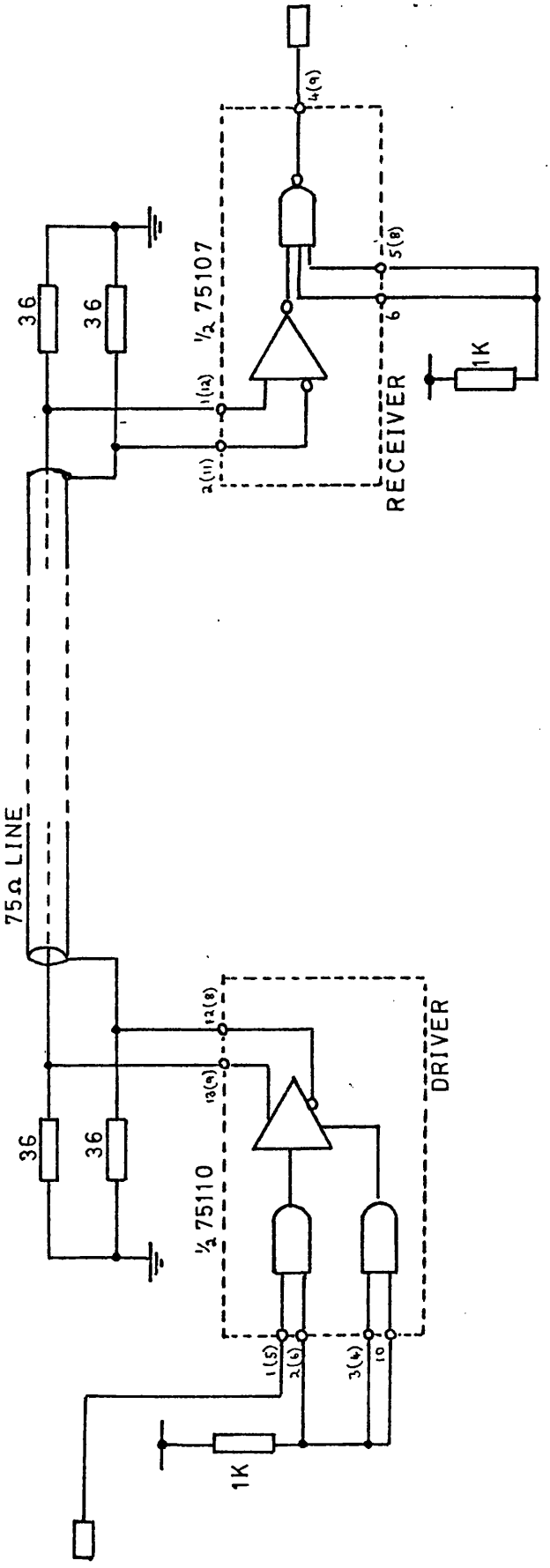


Fig. 4.1 Circuit Diagram of Line Driver/Receiver Card

(one channel)

4.3 UTILIZATION

4.3.1 Input and Output Connections

4.3.1.1 Line Driver/Receiver Card in PDP-11/2Ø System

SOCKET J1 - Driver Input from PDP-11

Pin	Name	Signal Function
1	OUTØ1A	Auxilary Output Bit 1
2	OUTØØA	Auxilary Output Bit 0
3	DATA TMTD	Data Transmitted
4	NEW DATA RDY	New Data Ready
5	OUT15	Output Bit 15
6	OUT14	" " 14
7	OUT13	" " 13
8	OUT12	" " 12
9	OUT11	" " 11
1Ø	OUT1Ø	" " 1Ø
11	OUTØ9	" " 9
12	OUTØ8	" " 8
13	OUTØ7	" " 7
14	OUTØ6	" " 6
15	OUTØ5	" " 5
16	OUTØ4	" " 4
17	OUTØ3	" " 3
18	OUTØ2	" " 2
19	OUTØ1	" " 1
2Ø	OUTØØ	" " Ø

SOCKET J2 - Receiver Output to PDP-11

Pin	Name	Signal Function
1	INØ1A	Auxilary Input Bit 1
2	INØØA	" " " Ø
3	REQ A	Interrupt Request A
4	REQ B	" " B
5	IN15	Input Bit 15
6	IN14	" " 14
7	IN13	" " 13
8	IN12	" " 12
9	IN11	" " 11
1Ø	IN1Ø	" " 1Ø
11	INØ9	" " 9
12	INØ8	" " 8
13	INØ7	" " 7
14	INØ6	" " 6

Pin	Name	Signal Function
15	INØ5	" " 5
16	INØ4	" " 4
17	INØ3	" " 3
18	INØ2	" " 2
19	INØ1	" " 1
2Ø	INØØ	" " Ø

EDGE CONNECTOR P1 - Line Inputs and Outputs

Pin	Name	Signal Function	Cable Colour
1		Not Used	
2		Not Used	
3	GND	Supply Common	
4	GND	" "	
5	$\overline{\text{OUTØØ}}$	Output Bit Ø (B)	White-Blue
6	OUTØØ	" " Ø (A)	Blue
7	$\overline{\text{OUTØ1}}$	" " 1 (B)	White-Blue
8	OUTØ1	" " 1 (A)	Orange
9	$\overline{\text{OUTØ2}}$	" " 2 (B)	White-Blue
1Ø	OUTØ2	" " 2 (A)	Green
11	$\overline{\text{OUTØ3}}$	" " 3 (B)	White-Blue
12	OUTØ3	" " 3 (A)	Brown
13	$\overline{\text{OUTØ4}}$	" " 4 (B)	White-Blue
14	OUTØ4	" " 4 (A)	Grey
15	$\overline{\text{OUTØ5}}$	" " 5 (B)	Red-Blue
16	OUTØ5	" " 5 (A)	Blue
17	$\overline{\text{OUTØ6}}$	" " 6 (B)	Red-Blue
18	OUTØ6	" " 6 (A)	Orange
19	$\overline{\text{OUTØ7}}$	" " 7 (B)	Red-Blue
2Ø	OUTØ7	" " 7 (A)	Green
21	$\overline{\text{OUTØ8}}$	" " 8 (B)	Red-Blue
22	OUTØ8	" " 8 (A)	Brown
23	$\overline{\text{OUTØ9}}$	" " 9 (B)	Red-Blue
24	OUTØ9	" " 9 (A)	Grey
25	$\overline{\text{OUT1Ø}}$	" " 1Ø (B)	Blue-Black
26	OUT1Ø	" " 1Ø (A)	Blue
27	$\overline{\text{OUT11}}$	" " 11 (B)	Blue-Black
28	OUT11	" " 11 (A)	Orange
29	$\overline{\text{OUT12}}$	" " 12 (B)	Blue-Black
3Ø	OUT12	" " 12 (A)	Green
31	$\overline{\text{OUT13}}$	" " 13 (B)	Blue-Black
32	OUT13	" " 13 (A)	Brown
33	$\overline{\text{OUT14}}$	" " 14 (B)	Blue-Black
34	OUT14	" " 14 (A)	Grey
35	$\overline{\text{OUT15}}$	" " 15 (B)	Yellow-Blue
36	OUT15	" " 15 (A)	Blue
37	$\overline{\text{NEW DATA RDY}}$	New Data Ready (B)	Yellow-Blue
38	NEW DATA READY	" " " (A)	Orange
39	$\overline{\text{DATA TMTD}}$	Data Transmitted (B)	Yellow-Blue
4Ø	DATA TMTD	" " (A)	Green

41	OUTØØA	Auxilary Output Bit Ø (B)	Yellow-Blue
42	OUTØØA	" " " Ø (A)	Brown
43	OUTØ1A	" " " 1 (B)	Yellow-Blue
44	OUTØ1A	" " " 1 (A)	Grey
45		Not Used	
46		" "	
47	-9VDC	-9 volt Source Power	
48	-9VDC	-9 " " "	
49	-5VDC	-5 volt Source Power	
5Ø	-5VDC	-5 " " "	
51		Not Used	
52		" "	
53	-12VDC	-12 volt Source Power	
54	-12VDC	-12 " " "	
55	+12VDC	+12 volt Source Power	
56	+12VDC	+12 " " "	
57		Not Used	
58		" "	
59	INØØ	Input Bit Ø (B)	White
6Ø	INØØ	" " Ø (A)	Blue
61	INØ1	" " 1 (B)	White
62	INØ1	" " 1 (A)	Orange
63	INØ2	" " 2 (B)	White
64	INØ2	" " 2 (A)	Green
65	INØ3	" " 3 (B)	White
66	INØ3	" " 3 (A)	Brown
67	INØ4	" " 4 (B)	White
68	INØ4	" " 4 (A)	Grey
69	INØ5	" " 5 (B)	Red
7Ø	INØ5	" " 5 (A)	Blue
71	INØ6	" " 6 (B)	Red
72	INØ6	" " 6 (A)	Orange
73	INØ7	" " 7 (B)	Red
74	INØ7	" " 7 (A)	Green
75	INØ8	" " 8 (B)	Red
76	INØ8	" " 8 (A)	Brown
77	INØ9	" " 9 (B)	Red
78	INØ9	" " 9 (A)	Grey
79	IN1Ø	" " 1Ø (B)	Black
8Ø	IN1Ø	" " 1Ø (A)	Blue
81	IN11	" " 11 (B)	Black
82	IN11	" " 11 (A)	Orange
83	IN12	" " 12 (B)	Black
84	IN12	" " 12 (A)	Green
85	IN13	" " 13 (B)	Black
86	IN13	" " 13 (A)	Brown
87	IN14	" " 14 (B)	Black
88	IN14	" " 14 (A)	Grey
89	IN15	" " 15 (B)	Yellow
9Ø	IN15	" " 15 (A)	Blue
91	REQ B	Interrupt Request B (B)	Yellow
92	REQ B	" " B (A)	Orange
93	REQ A	" " A (B)	Yellow
94	REQ A	" " A (A)	Green
95	INØØA	Auxilary Input Bit Ø (B)	Yellow
96	INØØA	" " " Ø (A)	Brown
97	INØ1A	" " " 1 (B)	Yellow
98	INØ1A	" " " 1 (A)	Grey

99	+5VDC	+5 volt Source Power
1ØØ	+5VDC	+5 " " "

4.3.1.2 Line Driver/Receiver Card in I8Ø8Ø System

SOCKET J1 - Driver Input from Interface

Pin	Name	Signal Function
1	INØ1A	Auxilary Input Bit 1
2	INØØA	" " " Ø
3	REQ A	Interrupt Request A
4	REQ B	" " B
5	IN15	Input Bit 15
6	IN14	" " 14
7	IN13	" " 13
8	IN12	" " 12
9	IN11	" " 11
1Ø	IN1Ø	" " 1Ø
11	INØ9	" " 9
12	INØ8	" " 8
13	INØ7	" " 7
14	INØ6	" " 6
15	INØ5	" " 5
16	INØ4	" " 4
17	INØ3	" " 3
18	INØ2	" " 2
19	INØ1	" " 1
2Ø	INØØ	" " Ø

SOCKET J2 - Receiver Output to Interface

Pin	Name	Signal Function
1	OUTØ1A	Auxilary Output Bit 1
2	OUTØØA	" " " Ø
3	DATA TMTD	Data Transmitted
4	NEW DATA RDY	New Data Ready
5	OUT15	Output Bit 15
6	OUT14	" " 14
7	OUT13	" " 13
8	OUT12	" " 12
9	OUT11	" " 11
1Ø	OUT1Ø	" " 1Ø
11	OUTØ9	" " 9
12	OUTØ8	" " 8
13	OUTØ7	" " 7
14	OUTØ6	" " 6
15	OUTØ5	" " 5
16	OUTØ4	" " 4
17	OUTØ3	" " 3
18	OUTØ2	" " 2
19	OUTØ1	" " 1
2Ø	OUTØØ	" " Ø

EDGE CONNECTOR P1 - Line Inputs and Outputs

Pin	Name	Signal Function	Cable Colour
1		Not Used	
2		Not Used	
3	GND	Supply Common	
4	GND	" "	
5	IN00	Input Bit 0 (B)	White
6	IN00	" " 0 (A)	Blue
7	IN01	" " 1 (B)	White
8	IN01	" " 1 (A)	Orange
9	IN02	" " 2 (B)	White
10	IN02	" " 2 (A)	Green
11	IN03	" " 3 (B)	White
12	IN03	" " 3 (A)	Brown
13	IN04	" " 4 (B)	White
14	IN04	" " 4 (A)	Grey
15	IN05	" " 5 (B)	Red
16	IN05	" " 5 (A)	Blue
17	IN06	" " 6 (B)	Red
18	IN06	" " 6 (A)	Orange
19	IN07	" " 7 (B)	Red
20	IN07	" " 7 (A)	Green
21	IN08	" " 8 (B)	Red
22	IN08	" " 8 (A)	Brown
23	IN09	" " 9 (B)	Red
24	IN09	" " 9 (A)	Grey
25	IN10	" " 10 (B)	Black
26	IN10	" " 10 (A)	Blue
27	IN11	" " 11 (B)	Black
28	IN11	" " 11 (A)	Orange
29	IN12	" " 12 (B)	Black
30	IN12	" " 12 (A)	Green
31	IN13	" " 13 (B)	Black
32	IN13	" " 13 (A)	Brown
33	IN14	" " 14 (B)	Black
34	IN14	" " 14 (A)	Grey
35	IN15	" " 15 (B)	Yellow
36	IN15	" " 15 (A)	Blue
37	REQ B	Interrupt Request B (B)	Yellow
38	REQ B	" " B (A)	Orange
39	REQ A	" " A (B)	Yellow
40	REQ A	" " A (A)	Green
41	IN00A	Auxilary Input Bit 0 (B)	Yellow
42	IN00A	" " " 0 (A)	Brown
43	IN01A	" " " 1 (B)	Yellow
44	IN01A	" " " 1 (A)	Grey
45		Not Used	
46		" "	
47	-9VDC	-9 volt Source Power	
48	-9VDC	-9 " " "	
49	-5VDC	-5 " " "	
50	-5VDC	-5 " " "	

51		Not Used				
52		" "				
53	-12VDC	-12 volt Source Power				
54	-12VDC	-12 " " "				
55	+12VDC	+12 " " "				
56	+12VDC	+12 " " "				
57		Not Used				
58		" "				
59	$\overline{\text{OUTØØ}}$	Output Bit Ø	(B)			White-Blue
6Ø	OUTØØ	" " Ø	(A)			Blue
61	$\overline{\text{OUTØ1}}$	" " 1	(B)			White-Blue
62	OUTØ1	" " 1	(A)			Orange
63	$\overline{\text{OUTØ2}}$	" " 2	(B)			White-Blue
64	OUTØ2	" " 2	(A)			Green
65	$\overline{\text{OUTØ3}}$	" " 3	(B)			White-Blue
66	OUTØ3	" " 3	(A)			Brown
67	$\overline{\text{OUTØ4}}$	" " 4	(B)			White-Blue
68	OUTØ4	" " 4	(A)			Grey
69	$\overline{\text{OUTØ5}}$	" " 5	(B)			Red-Blue
7Ø	OUTØ5	" " 5	(A)			Blue
71	$\overline{\text{OUTØ6}}$	" " 6	(B)			Red-Blue
72	OUTØ6	" " 6	(A)			Orange
73	$\overline{\text{OUTØ7}}$	" " 7	(B)			Red-Blue
74	OUTØ7	" " 7	(A)			Green
75	$\overline{\text{OUTØ8}}$	" " 8	(B)			Red-Blue
76	OUTØ8	" " 8	(A)			Brown
77	$\overline{\text{OUTØ9}}$	" " 9	(B)			Red-Blue
78	OUTØ9	" " 9	(A)			Grey
79	$\overline{\text{OUT1Ø}}$	" " 1Ø	(B)			Blue-Black
8Ø	OUT1Ø	" " 1Ø	(A)			Blue
81	$\overline{\text{OUT11}}$	" " 11	(B)			Blue-Black
82	OUT11	" " 11	(A)			Orange
83	$\overline{\text{OUT12}}$	" " 12	(B)			Blue-Black
84	OUT12	" " 12	(A)			Green
85	$\overline{\text{OUT13}}$	" " 13	(B)			Blue-Black
86	OUT13	" " 13	(A)			Brown
87	$\overline{\text{OUT14}}$	" " 14	(B)			Blue-Black
88	OUT14	" " 14	(A)			Grey
89	$\overline{\text{OUT15}}$	" " 15	(B)			Yellow-Blue
9Ø	OUT15	" " 15	(A)			Blue
91	$\overline{\text{NEW DATA RDY}}$	New Data Ready	(B)			Yellow-Blue
92	NEW DATA RDY	" " "	(A)			Orange
93	$\overline{\text{DATA TMTD}}$	Data Transmitted	(B)			Yellow-Blue
94	DATA TMTD	" "	(A)			Green
95	$\overline{\text{OUTØØA}}$	Auxilary Output Bit Ø		(B)		Yellow-Blue
96	OUTØØA	" " " Ø		(A)		Brown
97	$\overline{\text{OUTØ1A}}$	" " " 1		(B)		Yellow-Blue
98	OUTØ1A	" " " 1		(A)		Grey
99	+5VDC	+5 volt Source Power				
1ØØ	+5VDC	+5 " " "				

4.3.2 Transmission Link Cable Connections

I8Ø8Ø to PDP-11/2Ø

Pair No	1st Wire	(Pin No)	2nd Wire	(Pin No)	Signal
1	White	(1)	Blue	(2)	INØØ
2	White	(3)	Orange	(4)	INØ1
3	White	(5)	Green	(6)	INØ2
4	White	(7)	Brown	(8)	INØ3
5	White	(9)	Grey	(1Ø)	INØ4
6	Red	(11)	Blue	(12)	INØ5
7	Red	(13)	Orange	(14)	INØ6
8	Red	(15)	Green	(16)	INØ7
9	Red	(17)	Brown	(18)	INØ8
1Ø	Red	(19)	Grey	(2Ø)	INØ9
11	Black	(21)	Blue	(22)	IN1Ø
12	Black	(23)	Orange	(24)	IN11
13	Black	(25)	Green	(26)	IN12
14	Black	(27)	Brown	(28)	IN13
15	Black	(29)	Grey	(3Ø)	IN14
16	Yellow	(31)	Blue	(32)	IN15
17	Yellow	(33)	Orange	(34)	REQ B
18	Yellow	(35)	Green	(36)	REQ A
19	Yellow	(37)	Brown	(38)	
2Ø	Yellow	(39)	Grey	(4Ø)	

PDP-11/2Ø to I8Ø8Ø

Pair No	1st Wire (Pin No)	2nd Wire (Pin No)	Signal
21	White-Blue (1)	Blue (2)	OUTØØ
22	White-Blue (3)	Orange (4)	OUTØ1
23	White-Blue (5)	Green (6)	OUTØ2
24	White-Blue (7)	Brown (8)	OUTØ3
25	White-Blue (9)	Grey (10)	OUTØ4
26	Red-Blue (11)	Blue (12)	OUTØ5
27	Red-Blue (13)	Orange (14)	OUTØ6
28	Red-Blue (15)	Green (16)	OUTØ7
29	Red-Blue (17)	Brown (18)	OUTØ8
3Ø	Red-Blue (19)	Grey (2Ø)	OUTØ9
31	Blue-Black (21)	Blue (22)	OUT1Ø
32	Blue-Black (23)	Orange (24)	OUT11
33	Blue-Black (25)	Green (26)	OUT12
34	Blue-Black (27)	Brown (28)	OUT13
35	Blue-Black (29)	Grey (3Ø)	OUT14
36	Yellow-Blue(31)	Blue (32)	OUT15
37	Yellow-Blue(33)	Orange (34)	NEW DATA RDY
38	Yellow-Blue(35)	Green (36)	DATA TMTD
39	Yellow-Blue(37)	Brown (38)	
4Ø	Yellow-Blue(39)	Grey (4Ø)	

NB. In a 'Colour A-Colour B' combination, the first colour is the base colour.

4.3.3 Installation Data

Connector to Mother Card:-

Dual 5Ø-pin PC Edge Connector. Ø.125" diameter centres
(Compatible with SAE type C8ØØ1ØØ
or CDC type VPBØ1CSØEØØA1
or Viking Industries 3VH5Ø/1CNS

Connector to Interface Card or PDP-11/20 General Device Interface:-

20-pin SIL socket 0.1" centres
(Compatible with Jermyn type A23-2074PS)

Power Supply Requirements:-

+5v d.c. ±5% @ 0.4 A typ. (0.7 A max)
-5v d.c. ±5% @ 0.4 A typ. (0.7 A max)

Operating temperature range:-

+0°C to +70°C

SECTION 5

MICROPROCESSOR CONTROL PANEL AND DISPLAY PORT

Control of all microprocessor functions is available from the PDP-11/2Ø computer via the Interface Card and transmission link (Sections 2,3 and 4). However, during the system development stage, it is useful to have a means of controlling the microprocessor directly and to check some of the basic microprocessor operations. This facility is provided by the microprocessor control panel.

Similarly, it is useful to have a programmable display port in a location near to the micro-machine system such that program flow conditions, error states, etc can be displayed as part of the immediate control system. This is provided by the Data Display Card.

5.1 FUNCTION DESCRIPTION

5.1.1 Microprocessor Control Panel

The microprocessor control panel has three switches which function as follows:-

RESET

General RESET of the microprocessor and its associated peripherals.

The microprocessor's internal program counter and instruction register are reset to zero such that program execution will recommence at memory location zero.

The Interface address register and data transfer flags are reset cancelling any transfers in progress. All peripheral flags are reset and any outstanding peripheral interrupts are cleared from the hardware stack.

The mode of operation of the peripherals is not altered (Sections 7 - 11), i.e. any

peripheral which is interrupt enabled will remain in that state unless a software routine clears it.

WAIT — Causes the microprocessor to enter a WAIT state following the next T_2 machine cycle. The processor will remain in this state as long as the switch is on unless the S.STEP switch is used.

S.STEP — Causes the microprocessor to SINGLE STEP to the next machine cycle. Only effective in the WAIT state caused by the control panel WAIT switch (see Note below).

Note: As the $\overline{\text{RESET}}$ and $\overline{\text{WAIT REQ}}$ control lines of the microprocessor bus are driven by open-collector logic from both the Control Panel and the Interface Card, then some conflict may occur in use. Generally, no conflict can occur in the use of the RESET control as this is caused by either device temporarily pulling the $\overline{\text{RESET}}$ line low which will initiate the correct action. However, if more than one device has requested a WAIT state then the SINGLE STEP mode will be inoperative from any device as the other(s) will still cause the processor to wait at its current machine cycle (see Section 5.2 for details).

The Control Panel display lights indicate the basic microprocessor states as follows:-

RUN — The RUN light is on when the microprocessor system is active. This means that it is either running a program under its own control or it is in a direct memory access mode (DMA) from the PDP-11 (currently the only device with DMA capability). In the latter case the HOLD light will also be on.

WAIT — The WAIT light is on whenever the microprocessor enters a WAIT state. This may be

caused by the WAIT switch (above), an Interface Card command (Section 2.2.5) or a slow memory response. The latter case will cause the WAIT light to glow dimly during normal program running as the standard memory card has an access time of 68Ø nS causing a wait of one clock cycle per machine cycle in memory access operations.

HOLD — The HOLD light is on whenever the microprocessor has freed its address and data busses for a DMA operation. The RUN light will also be on to indicate the system is active.

HALT — The HALT light is on whenever the microprocessor has executed a 'HLT' instruction and is thus no longer running a program.

INT DIS — The INT DIS (Interrupt Disabled) light indicates that the microprocessor CPU has disabled its internal interrupt flag. Thus interrupts from all external devices (including the PDP-11) will be ignored although they will be held in the hardware stack (Section 7). The light directly indicates the state of the internal interrupt enable/disable flag and is thus 'on' following a RESET, an interrupt or a 'DI' instruction and is 'off' one instruction cycle after an 'EI' instruction.

5.1.2 Display Port

The display port consists of eight LED s normally connected to the Input/Output Card port no 3. This provides the facility to display general program data and status with a maximum of 256 coded options available. The output port complements the data

on output and this must be allowed for in the display routine, e.g. to display the contents of register B the following may be used:-

```
MOV  A,B          ;MOVE INTO ACCUMULATOR
CMA               ;COMPLEMENT
OUT  3Q           ;OUTPUT TO PORT 3
```

The data display card has an auxiliary function in that it has an input socket, a flying output lead and bi-directional circuitry such that it may be inserted in series between any peripheral and its associated input/output port to temporarily display data transfers for debugging purposes.

5.2 CIRCUIT DESCRIPTION

5.2.1 Microprocessor Control Panel

The circuit diagram of the Microprocessor Control Panel is given in Fig 5.1. The RESET, WAIT and S.STEP switches are shown in their normal (off) position. The $\overline{\text{RESET}}$ and $\overline{\text{WAIT REQ}}$ bus lines are driven by open-collector logic gates A3 as other devices (notably the Interface Card) must also access these lines.

In the normal states, pin 12 of A3 will be held low by the WAIT switch and thus pin 11 will always be high, regardless of the other input, allowing the microprocessor to run and rendering the single-step circuitry inoperative. Pin 13 of A3 will normally be held high by its biasing resistor and thus if the WAIT switch is opened pin 12 will also go high causing pin 11 to go low and thus issuing a $\overline{\text{WAIT REQ}}$ to the microprocessor. Assuming the microprocessor CPU is free to honour this request then a WAIT state will be entered following the T_2 state of the next machine cycle (see reference 67 for further details). While the microprocessor is in the WAIT state, a momentary depression of the S.STEP switch will cause pin 5 of A3 to go high and thus its output to go low. The cross-coupling of the gates of A3 acts as a bounce-suppression circuit. When pin 6 of A3 goes low then pin 13 of A3 is temporarily pulled low by the resistor-capacitor

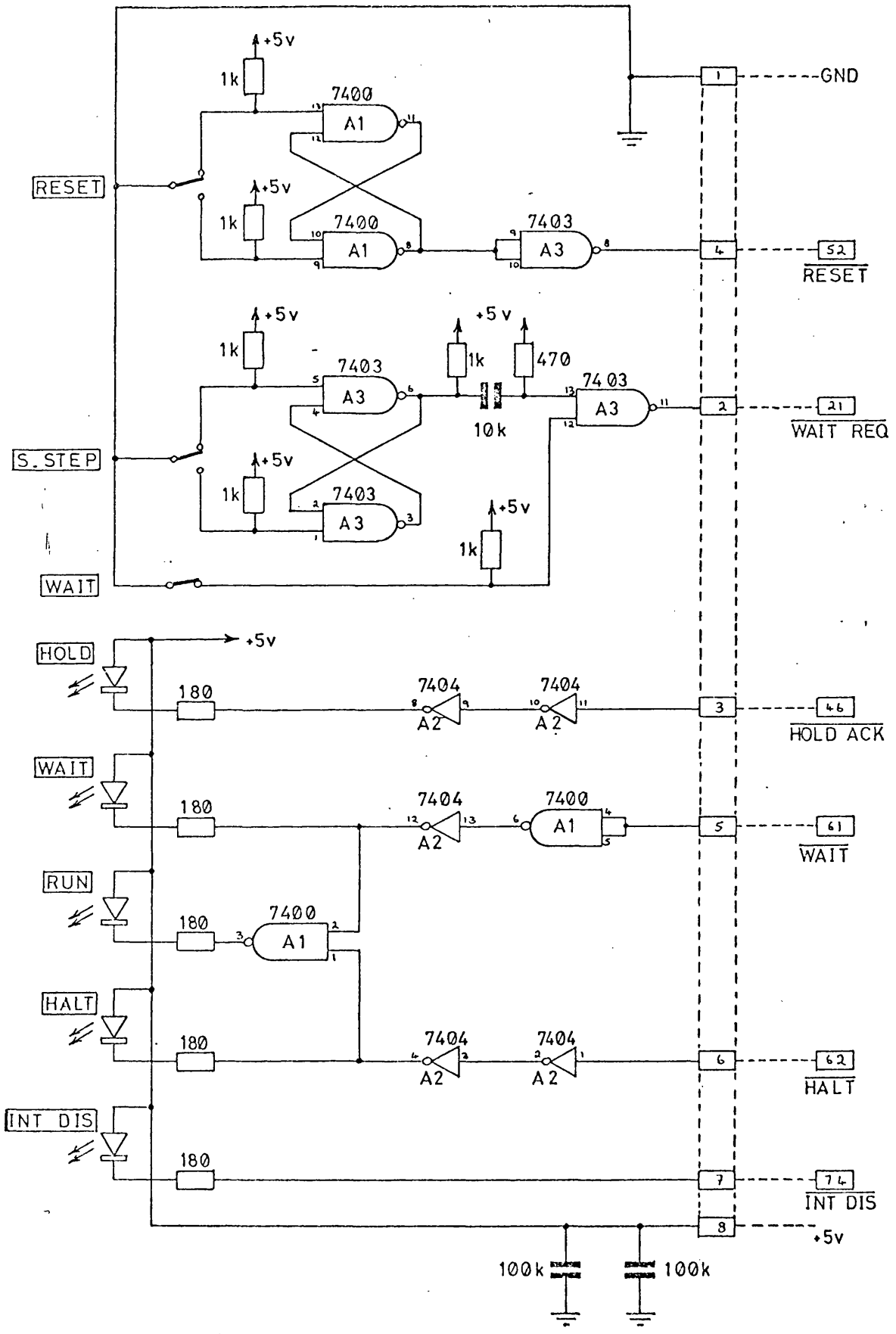


Fig. 5.1 Microprocessor Control Panel Circuit Diagram

network which causes the output, pin 11, to go high for approximately 1.5 microseconds. Assuming no other device is holding the $\overline{\text{WAIT REQ}}$ line low, this will allow the microprocessor to continue its current machine cycle but will again cause a WAIT state to occur following the next T_2 state. Thus the processor will single-step through its operating program.

Operation of the RESET switch is also 'cleaned-up' by a pair of bounce-suppression gates in A1. A temporary depression of the RESET switch will cause pin 8 of A1 to go high which is inverted in the 8, 9, 1Ø section of A3 to cause the $\overline{\text{RESET}}$ bus to be pulled low with the consequences described in Section 5.1.1.

The WAIT, HOLD, HALT and INT DIS display lights are driven directly or via suitable buffering from the appropriate microprocessor bus line. The exception is the RUN light whose drive signal is the output from pin 3 of the NAND gate A1. The inputs, pins 1 and 2, are the $\overline{\text{WAIT}}$ and $\overline{\text{HALT}}$ signals such that the RUN light will be on in all states other than a WAIT or following a 'HLT' instruction.

5.2.2 Data Display Card

The circuit diagram of the Data Display Card is given in Fig 5.2. The links between the plug P1 and the socket J1 are bi-directional such that the card may be inserted in series between any peripheral and its associated input or output port to visually indicate the transfers in progress. The display LED s are supplied from the +5vdc supply rail and driven via inverters such that each LED will light when its associated data line at J1 or P1 is high (logical '1'). It should be noted that the actual input and output ports of the I8Ø8Ø complement the data during the I/O operation.

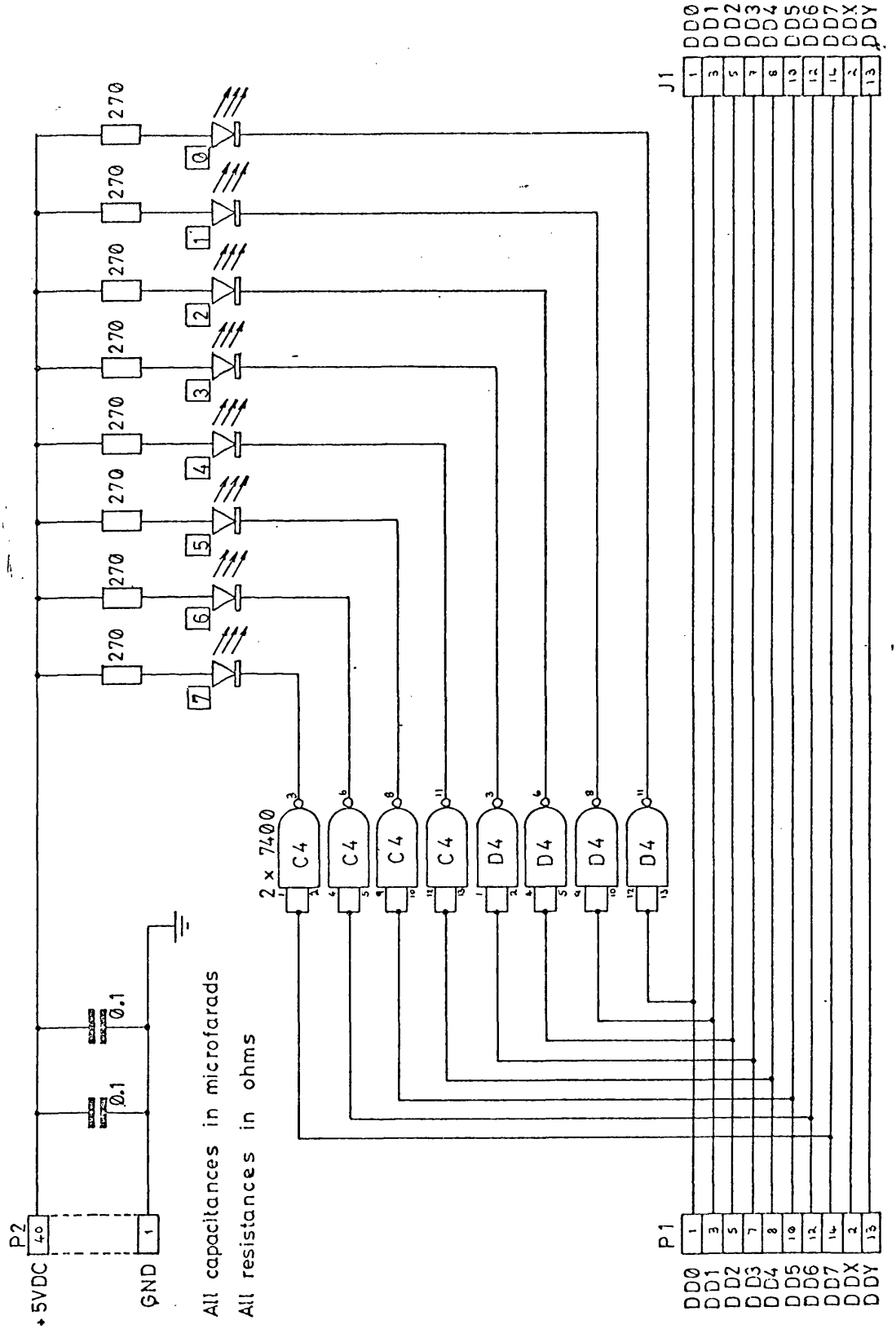


Fig. 5.2 Data Display Card Circuit Diagram

5.3 UTILISATION

5.3.1 Microprocessor Control Panel

5.3.1.1 Connections List

Pin	Name	Signal Function	Bus Pin No
1	GND	Supply Common	3, 4
2	WAIT REQ	Ready flag from memory	21
3	HOLD ACK	Acknowledge Hold Request	46
4	RESET	Initiate External Reset	52
5	WAIT	Wait Request Acknowledge	61
6	HALT	Halt Cycle Status	62
7	INT DIS	Interrupt Disabled Flag	74
8	+5 VDC	V_{CC} Source Power	99, 1ØØ

5.3.1.2 Installation Data

Connector to microprocessor bus:-

Solder connected to microprocessor bus

Power Supply Requirements:-

+5 vdc ±5% @ 1ØØ mA typ (16Ø mA max)

Operating Temperature Range:-

+Ø°C to +7Ø°C

5.3.2 Data Display Card

5.3.2.1 Connections List

SOCKET J1 - Data Input/Output Socket

Pin	Name	Signal Function
1	DDØ	Data Display Bit Ø
2	DDX	" " Auxilary Bit X
3	DD1	" " Bit 1
4		Not Used
5	DD2	Data Display Bit 2
6		Not Used
7	DD3	Data Display Bit 3
8	DD4	" " " 4
9		Not Used
1Ø	DD5	Data Display Bit 5
11		Not Used
12	DD6	Data Display Bit 6
13	DDY	" " Auxilary Bit Y
14	DD7	" " Bit 7

PLUG P1 - Data Output/Input Plug

Pin	Name	Signal Function
1	DDØ	Data Display Bit Ø
2	DDX	" " Auxilary Bit 'X'
3	DD1	" " Bit 1
4		Not Used
5	DD2	Data Display Bit 2
6		Not Used
7	DD3	Data Display Bit 3
8	DD4	" " " 4
9		Not Used
1Ø	DD5	Data Display Bit 5
11		Not Used
12	DD6	Data Display Bit 6
13	DDY	" " Auxilary Bit 'Y'
14	DD7	" " Bit 7

EDGE CONNECTOR P2 - Peripheral Bus Interconnector

Pin	Name	Signal Function
1	GND	Supply Common
2		Not Used
3		" "
4		" "
5		" "
6		" "
7		" "
8		" "
9		" "
1Ø		" "
11		" "
12		" "
13		" "
14		" "
15		" "
16		" "
17		" "
18		" "
19		" "
2Ø		" "
21		" "
22		" "
23		" "
24		" "
25		" "
26		" "
27		" "
28		" "
29		" "
3Ø		" "
31		" "
32		" "
33		" "
34		" "
35		" "
36		" "
37		" "
38		" "
39		" "
4Ø	+5 VDC	+5 Volt Source Power

5.3.2.2 Installation Data

Connector to Peripheral Unit Bus:-

Single 4Ø-pin PC Edge Connector. Ø.1" centres
(Compatible with Vero type 13623-1)

Power Supply Requirements:-

+5 vdc ±5% @ 16Ø mA typ. (2ØØ mA max)

Operating Temperature Range:-

+Ø°C to +7Ø°C

SECTION 6

INPUT/OUTPUT CARD

The Input/Output Card has been designed to provide an input/output facility containing eight individually addressable input ports and four individually addressable output ports on each card. The card contains all logic necessary to support a multi-card implementation and is a plug-in replacement for the Intel imm 8-61 Input/Output Card.

In addition to its greater input/output capacity than the Intel equivalent, the Input/Output Card also contains data transfer control bits for each port such that hardware 'handshaking' modes of data transfer are possible. This facility greatly increases the speed and reliability of the operational micro-processor system.

6.1 GENERAL FUNCTIONAL DESCRIPTION

This section describes the operation of the Input/Output Card in general functional terms and is divided into sections which relate to the four functional units of the device:

1. The Module Decode Block
   This section determines which card is to be utilised for an operation when more than one card has been installed in the system.

2. The Port Decode Block
   This determines which of the 256 possible input and output ports is to be used for an operation.

3. The Input Block
   This block contains the eight input ports and their associated logic.

4. The Output Block
   This block contains the four output ports and their associated logic.

Fig 6.1 shows the block diagram of the Input/Output Card and indicates that each operation performed by an Input/Output Card must use functions 1, 2 and 3 or 1, 2 and 4 as described above.

6.1.1 Module and Port Select Operations

The first operation before any input/output takes place is always a module and port select operation. The sequence is as follows:

1. The CPU sends an I/O Address to the Module Select and Port Select blocks. This I/O Address contains the information necessary to specify which card is to be used for an operation (in a multi-card system), what type of operation is to be performed (Input or Output), and which port is to be used for that operation. Both the complemented and non-complemented levels of the high-order address lines are returned to the mother-board which, in turn, selectively returns either the complemented or non-complemented level of each of the high-order address bits (depending on the card position) to the module decoder on lines DS1Ø, 11, 14 and 15. Thus the position of a module determines which 32 addresses it will respond to.

2. The selected card is identified by the card's Module Address block which generates an enable signal which is then transmitted to the rest of the card's logic.

3. The Port-Decode Block on the selected card determines which of eight possible ports are being addressed (as bits 12 and 13 are fixed by hard-wiring) and then sends enabling signals to either the Input or Output block depending on which operation was requested.

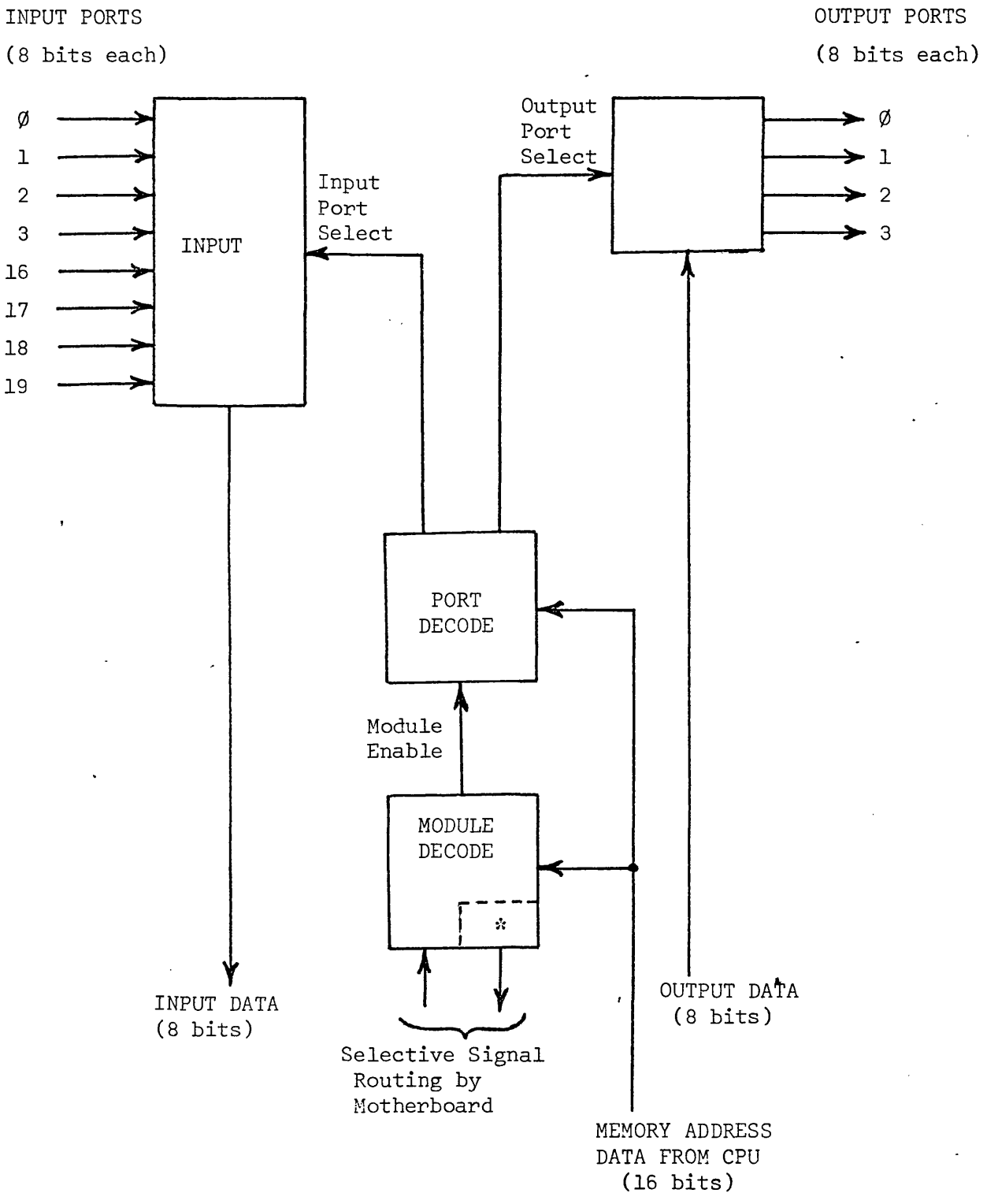


Fig 6.1 Input/Output Card Functional Block Diagram

6.1.2 Input Operation

The Input Operation brings eight bits of data from an external source and presents it to the CPU. An acknowledgement is sent to the sending device.

6.1.3 Output Operation

An Output Operation sends eight bits of data from the CPU to an external device. A request is sent to the external device to notify it that new data is ready (as this would not be evident if it was the same as the previous transmission) and the data is latched on the output port until another output operation to the same port is performed.

6.2 THEORY OF OPERATION

6.2.1 Module and Port Selection

The Input/Output Card circuit diagram is given in Fig 6.2.

Module address information is brought to the Input/Output Card edge pins; the module address is complemented by a series of inverting latches and the complemented address is present at additional edge pins. The user selects an address for each card and implements the address by selecting a set of Address and Complemented Address signals; selected signals are externally jumpered to the Module Selection circuits, which combine the incoming signals through a NAND gate (C3) to provide the enabling signal which is sent to other circuitry on the board.

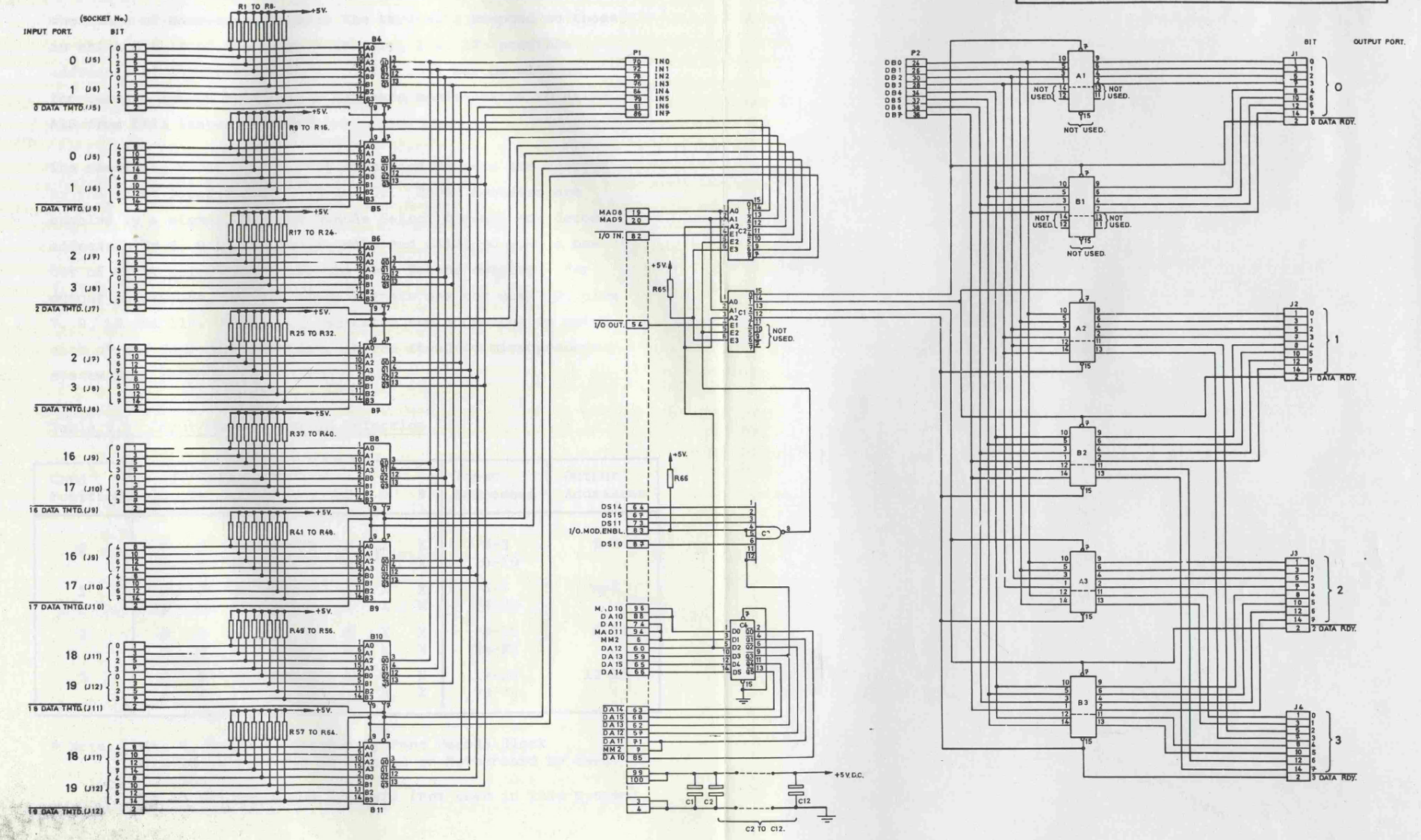
Addressing of an input/output port is performed using the high-order eight bits of the 16-bit address bus (bits 8 to 15 inclusive). The high-order six address lines are input through an inverting latch and both the complemented and non-complemented forms of these address bits are returned to the mother board. The mother board, in turn, selectively returns either the complemented or non-complemented forms of bits 1Ø, 11, 14 and 15

Fig. 6.2 Input/Output Card Circuit Diagram

CIRCUIT DIAGRAM

OF INPUT/OUTPUT CARD.

P.J.BURROWS. FEB. 1975.



(on lines DSlØ, DSll, DS14 and DS15) depending on the card position. These are input to the enabling NAND gate (A16). In the current version of the I/O Card, the complement of address line 13 is wired to an input of the NAND gate, A16, thereby reducing the range of addresses to which the card will respond to those in which Bit 13 of the address is zero, i.e. 128 possible addresses of input/output port. This range may be altered for certain boards by a simple hardware modification to drive A16 from DA13 instead of $\overline{DA13}$.

The remaining three bits of the specified address are decoded by the Input Decoder chips, C1 and C2. These decoders are enabled by a signal from the Module Select circuit and decode address bits 8, 9 and 12 (MAD8, MAD9 and DA12) to give a one out of eight select signal on the Port Decode outputs. For output operations, four of these signals are not used (C1 pins 7, 9, 1Ø and 11). Table 6.1 indicates the addresses to which each of the four card positions in the standard microprocessor system will respond.

Table 6.1 Input/Output Address Selection

Card Position	Address Bits * 15	14	13	12	11	1Ø	9	8	Input Addresses	Output Addresses
Ø	Ø	Ø	Ø	Ø	Ø	Ø	X	X	Ø-3	Ø-3
	Ø	Ø	Ø	1	Ø	Ø	X	X	16-19	
1	Ø	Ø	Ø	Ø	Ø	1	X	X	4-7	4-7
	Ø	Ø	Ø	1	Ø	1	X	X	2Ø-23	
2	Ø	Ø	Ø	Ø	1	Ø	X	X	8-11	8-11
	Ø	Ø	Ø	1	1	Ø	X	X	24-27	
3	Ø	Ø	Ø	Ø	1	1	X	X	12-15	12-15
	Ø	Ø	Ø	1	1	1	X	X	28-31	

* Note: Bits 8, 9 and 12 decoded by Port Decode Block
Bits 10, 11, 14 and 15 decoded on Mothercard by card position
Bit 13 hard-wired on I/O card (not used in this system)

6.2.2 Input Operations

During an input operation, the $\overline{\text{I/O IN}}$ control is held low by the CPU, causing the Port Decode chip C2 to be enabled on the selected card as detailed above. Thus, one of its eight output lines will go low. For example, if an input operation from port Ø is requested, then pin 15 of C2 (Fig 6.2) will go low which will enable the 'A' section inputs of multiplexers B4 and B5 to open passing the eight data bits from port Ø onto the common input data bus (INØ to IN7). At the same time, this select signal is also sent to pin 2 of J5 (the socket associated with port Ø) to act as a $\overline{\text{Ø DATA TMTD}}$ signal. This allows the sending peripheral to know that data has been read and to cease holding the data for transmission. However, the internal logic of the microprocessor system is such that this signal is a 'low' level pulse of duration 1.5 microseconds but a shorter pulse of 0.5 microseconds is also sent to other ports on the same card. If uncorrected, this would cause spurious acknowledgements to be sent to other peripherals having data for transfer to the microprocessor and so must be corrected by use of the simple circuit in Fig 6.3.

Fig 6.3 shows the Input Port Data Transfer Timing Circuit which is included as a part of each peripheral unit requiring a data transfer 'handshake'. The input at 'A' from the I/O Card $\overline{\text{DATA TMTD}}$ bit (pin 2) is timed in duration by the 74121 monostable circuit such that an output at 'D' is only obtained for a low level input of duration greater than 1.Ø microsecond.

6.2.3 Output Operations

An Output operation begins with the transmission of an I/O Address to the Input/Output Card from the CPU. The Module and Port Selection is performed as previously discussed, except that the $\overline{\text{I/O OUT}}$ control is now low and the port decoding is performed by the decoder chip C1. Thus, one of the four active outputs of the chip (pins 12, 13, 14 and 15) will go low to latch the data from the common data bus, DBØ to DB7, onto one of the four output ports formed by latches A1 to A3 and B1 to B3. This select

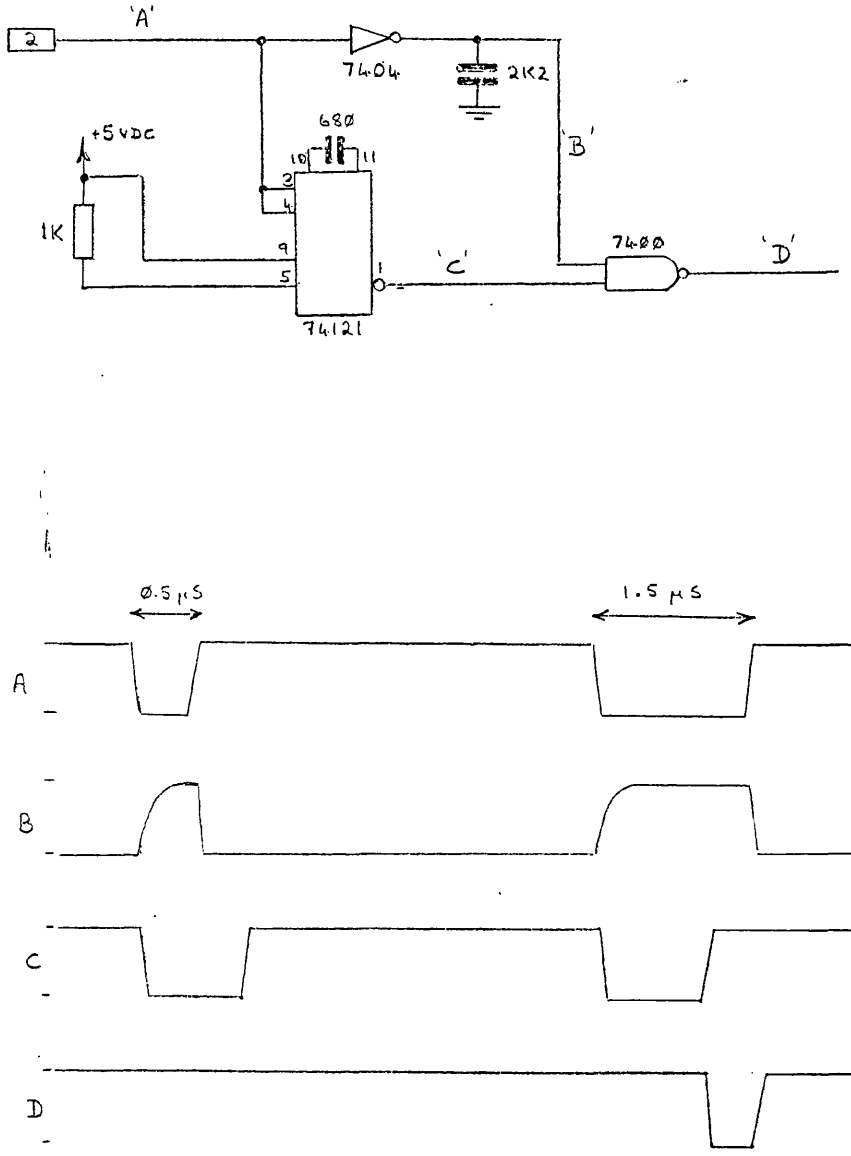


Fig 6.3 Input Port 'Handshake' Timing Circuit

signal is also sent to pin 2 of each output port to give a $\overline{\text{DATA RDY}}$ signal to any peripheral connected to that port so that it may be aware of the data transmission. In contrast to the input port, no additional timing circuitry is necessary for data transfer 'handshaking'.

6.3 UTILIZATION

6.3.1 Connection Lists

Note: All microprocessor input/output ports on in-house constructed cards have the following connection arrangement:

Pin	Name	Signal Function
1		Data Bit Ø
2		Data Transfer Control Bit (RDY or TMTD)
3		Data Bit 1
5		" " 2
7		" " 3
8		" " 4
1Ø		" " 5
12		" " 6
14		" " 7

Pins 4, 6, 9, 11 and 13 are unused on all input/output sockets and are thus not listed below.

INPUT/OUTPUT CARD NO Ø

SOCKET J1 - Data Output Port No Ø

Pin	Name	Signal Function
1	$\overline{\text{XIN0}}$	Auxilary Interface Input Bit Ø
2	$\overline{\text{XIN DATA RDY}}$	" " " Data Ready
3	$\overline{\text{XIN1}}$	" " " Bit 1
5	$\overline{\text{XIN2}}$	" " " " 2
7	$\overline{\text{XIN3}}$	" " " " 3
8	$\overline{\text{XIN4}}$	" " " " 4
1Ø	$\overline{\text{XIN5}}$	" " " " 5
12	$\overline{\text{XIN6}}$	" " " " 6
14	$\overline{\text{XIN7}}$	" " " " 7

SOCKET J2 - Data Output Port No 1

Pin	Name	Signal Function
1	$\overline{\text{VT STRT}}$	Start Terminal Voltage A/D Conversion
2	$\overline{\text{CNTRL1 DATA RDY}}$	Control Register No 1 Data Ready
3	$\overline{\text{EQ STRT}}$	Start E_q' Voltage A/D Conversion
5	$\overline{\text{VT INT ENBL}}$	Terminal Voltage A/D Interrupt Enable
7	$\overline{\text{EQ INT ENBL}}$	E_q' Voltage A/D Interrupt Enable
8	$\overline{\text{VT AUTO}}$	Terminal Voltage A/D Auto Conversion Mode
1Ø	$\overline{\text{EQ AUTO}}$	E_q' Voltage A/D Auto Conversion Mode
12	$\overline{\text{COMP CNTRL}}$	Computer Controlling Field Voltage
14	$\overline{\text{RESET PER}}$	Reset Microprocessor Peripherals

SOCKET J3 - Data Output Port No 2

Pin	Name	Signal Function
1	$\overline{\text{DELTA1 STRT}}$	Start Load Angle and Speed Conversion No 1
2	$\overline{\text{CNTRL2 DATA RDY}}$	Control Register No 2 Data Ready
3	$\overline{\text{DELTA2 STRT}}$	Start Load Angle and Speed Conversion No 2
5	$\overline{\text{DELTA INT ENBL}}$	Load Angle Transducer Interrupt Enable
7	$\overline{\text{DELTA AUTO}}$	Load Angle Conversion Auto Mode
8	$\overline{\text{VFI STRT}}$	Start Field Voltage A/D Conversion
1Ø	$\overline{\text{VFI INT ENBL}}$	Field Voltage A/D Interrupt Enable
12	$\overline{\text{VFI AUTO}}$	Field Voltage A/D Auto Conversion Mode
14	$\overline{\text{VT LOW INT ENBL}}$	Low Terminal Voltage Interrupt Enable

SOCKET J4 - Data Output Port No 3

Pin	Name	Signal Function
1	DDØ	Data Display Bit Ø
2	DD DATA RDY	" " Data Ready
3	DD1	" " Bit 1
5	DD2	" " " 2
7	DD3	" " " 3
8	DD4	" " " 4
1Ø	DD5	" " " 5
12	DD6	" " " 6
14	DD7	" " " 7

SOCKET J5 - Data Input Port No Ø

Pin	Name	Signal Function
1	$\overline{\text{XOUTØ}}$	Auxilary Interface Output Bit Ø
2	$\overline{\text{XOUT DATA TMTD}}$	" " " Data Transmitted
3	$\overline{\text{XOUT1}}$	" " " Bit 1
5	$\overline{\text{XOUT2}}$	" " " " 2

Pin	Name	Signal Function
7	XOUT3	Auxilary Interface Output Bit 3
8	XOUT4	" " " " 4
1Ø	XOUT5	" " " " 5
12	XOUT6	" " " " 6
14	XOUT7	" " " " 7

SOCKET J6 - Data Input Port No 1

Pin	Name	Signal Function
1	VT DONE	Terminal Voltage Transducer Done (Ready)
2		Not Used
3	EQ DONE	E'_q Transducer Done (ready)
5	DELTA1 DONE	Load Angle Transducer No 1 Done (ready)
7	DELTA2 DONE	" " " No 2 " (")
8	VFI DONE	Field Voltage Transducer Done (ready)
1Ø	AUX DONE	Spare " " " (")
12	PDP DATA RDY	Data Ready on Link from PDP-11 to I8Ø8Ø
14	PDP DONE	I8Ø8Ø to PDP-11 Data Transfer Done (ready)

SOCKET J7 - Data Input Port No 2

Pin	Name	Signal Function
1		Not Used
2		" "
3		" "
5		" "
7		" "
8		" "
1Ø		" "
12	VFIØ	Field Voltage Bit Ø (Ø.Ø1953 v)
14	VFI1	" " " 1 (+Ø.Ø39Ø6 v)

SOCKET J8 - Data Input Port No 3

Pin	Name	Signal Function
1	VFI2	Field Voltage Bit 2 (+Ø.Ø7813 v)
2	VFI DATA TMTD	" " Data Transmitted
3	VFI3	" " Bit 3 (+Ø.15626 v)
5	VFI4	" " " 4 (+Ø.3125 v)
7	VFI5	" " " 5 (+Ø.625 v)
8	VFI6	" " " 6 (+1.25 v)
1Ø	VFI7	" " " 7 (+2.5 v)
12	VFI8	" " " 8 (+5.Ø v)
14	VFI9	" " " 9 (-1Ø.Ø v)

SOCKET J9 - Data Input Port No 16

Pin	Name	Signal Function
1	SPAØ	Speed Count No 1 Bit Ø (1uS)
2		Not Used
3	SPA1	Speed Count No 1 Bit 1 (2uS)
5	SPA2	" " " " " 2 (4uS)
7	SPA3	" " " " " 3 (8uS)
8	SPA4	" " " " " 4 (16uS)
1Ø	SPA5	" " " " " 5 (32uS)
12	SPA6	" " " " " 6 (64uS)
14	SPA7	" " " " " 7 (128uS)

SOCKET J1Ø - Data Input Port No 17

Pin	Name	Signal Function
1	SPA8	Speed Count No 1 Bit 8 (Ø.256mS)
2		Not Used
3	SPA9	Speed Count No 1 Bit 9 (Ø.512mS)
5	SPA1Ø	" " " " " 1Ø (1.Ø24mS)
7	SPA11	" " " " " 11 (2.Ø48mS)
8	SPA12	" " " " " 12 (4.Ø96mS)
1Ø	SPA13	" " " " " 13 (8.192mS)
12	SPA14	" " " " " 14 (16.384mS)
14	SPA15	" " " " " 15 (-32.768mS)

SOCKET J11 - Data Input Port No 18

Pin	Name	Signal Function
1	$\overline{\text{DELAØ}}$	Shaft Angle No 1 Bit Ø (1uS)
2		Not Used
3	$\overline{\text{DELA1}}$	Shaft Angle No 1 Bit 1 (2uS)
5	$\overline{\text{DELA2}}$	" " " " " 2 (4uS)
7	$\overline{\text{DELA3}}$	" " " " " 3 (8uS)
8	$\overline{\text{DELA4}}$	" " " " " 4 (16uS)
1Ø	$\overline{\text{DELA5}}$	" " " " " 5 (32uS)
12	$\overline{\text{DELA6}}$	" " " " " 6 (64uS)
14	$\overline{\text{DELA7}}$	" " " " " 7 (128uS)

SOCKET J12 - Data Input Port No 19

Pin	Name	Signal Function
1	$\overline{\text{DELA8}}$	Shaft Angle No 1 Bit 8 (Ø.256mS)
2	$\overline{\text{DELA DATA TMTD}}$	" " " " Data Transmitted
3	$\overline{\text{DELA9}}$	" " " " Bit 9 (Ø.512mS)
5	$\overline{\text{DELA1Ø}}$	" " " " " 1Ø (1.Ø24mS)

Pin	Name	Signal Function
7	$\overline{\text{DELA11}}$	Shaft Angle No 1 Bit 11 (2.Ø48mS)
8	$\overline{\text{DELA12}}$	" " " " " 12 (4.Ø96mS)
1Ø	$\overline{\text{DELA13}}$	" " " " " 13 (8.192mS)
12	$\overline{\text{DELA14}}$	" " " " " 14 (16.384mS)
14	$\overline{\text{DELA15}}$	" " " " " 15 (-32.768mS)

EDGE CONNECTOR P1 - Microprocessor Bus Connector

Pin	Name	Signal Function
1		Not Used
2		" "
3	GND	Supply Common
4	GND	" "
5		Not Used
6	MM2	Auxilary Address Bit
7	$\overline{\text{MM2}}$	" " " (Complement)
8		Not Used
9		" "
1Ø		" "
11		" "
12		" "
13		" "
14		" "
15		" "
16		" "
17		" "
18		" "
19	MAD8	Address Bit 8
2Ø	MAD9	" " 9
21		Not Used
22		" "
23		" "
24	DBØ	Output Data Bit Ø
25		Not Used
26	DB1	Output Data Bit 1
27		Not Used
28	DB3	Output Data Bit 3
29		Not Used
3Ø	DB2	Output Data Bit 2
31		Not Used
32	DB5	Output Data Bit 5
33		Not Used
34	DB4	Output Data Bit 4
35		Not Used
36	DB7	Output Data Bit 7
37		Not Used
38	DB6	Output Data Bit 6
39		Not Used
4Ø		" "
41		" "
42		" "
43		" "

Pin	Name	Signal Function
44		Not Used
45		" "
46		" "
47		" "
48		" "
49		" "
5Ø		" "
51		" "
52		" "
53		" "
54	$\overline{\text{I/O OUT}}$	I/O Output Strobe
55		Not Used
56		" "
57	$\overline{\text{DA12}}$	Device Address Bit 12 (Complement)
58		Not Used
59	DA13	Device Address Bit 13
6Ø	DA12	" " " 12
61		Not Used
62	$\overline{\text{DA13}}$	Device Address Bit 13 (Complement)
63	$\overline{\text{DA14}}$	" " " 14 (")
64	DS14	" Select Bit 14
65	DA15	" Address Bit 15
66	DA14	" " " 14
67	$\overline{\text{DS15}}$	" Select Bit 15
68	$\overline{\text{DA15}}$	" Address Bit 15 (Complement)
69		Not Used
7Ø	$\overline{\text{INØ}}$	Peripheral Input Bit Ø
71	$\overline{\text{DA11}}$	Device Address Bit 11 (Complement)
72	IN1	Peripheral Input Bit 1
73	DS11	Device Select Bit 11
74	DA11	" Address Bit 11
75		Not Used
76	IN3	Peripheral Input Bit 3
77		Not Used
78	IN2	Peripheral Input Bit 2
79	IN5	" " " 5
8Ø		Not Used
81	$\overline{\text{IN6}}$	Peripheral Input Bit 6
82	$\overline{\text{I/O IN}}$	I/O Input Strobe
83	I/O MOD ENBL	I/O Module Enable
84	$\overline{\text{IN4}}$	Peripheral Input Bit 4
85	$\overline{\text{DA1Ø}}$	Device Address Bit 1Ø (Complement)
86	IN7	Peripheral Input Bit 7
87	DS1Ø	Device Select Bit 1Ø
88	DA1Ø	" Address Bit 1Ø
89		Not Used
9Ø		" "
91		" "
92		" "
93		" "
94	MAD11	Address Bit 11
95		Not Used
96	MAD1Ø	Address Bit 1Ø
97		Not Used
98		" "
99	+5VDC	+5 volt Source Power
1ØØ	+5VDC	+5 " " "

SOCKET J1 - Data Output Port No 4

Pin	Name	Signal Function
1		Not Used
2		" "
3		" "
5		" "
7		" "
8	$\overline{VF0}$	Field Voltage Bit Ø (+Ø.ØØ488v)
1Ø	$\overline{VF1}$	" " " 1 (+Ø.ØØ977v)
12	$\overline{VF2}$	" " " 2 (+Ø.Ø1953v)
14	$\overline{VF3}$	" " " 3 (+Ø.Ø39Ø6v)

SOCKET J2 - Data Output Port No 5

Pin	Name	Signal Function
1	$\overline{VF4}$	Field Voltage Bit 4 (+Ø.Ø7813v)
2		Not Used
3	$\overline{VF5}$	Field Voltage Bit 5 (+Ø.15625v)
5	$\overline{VF6}$	" " " 6 (+Ø.3125v)
7	$\overline{VF7}$	" " " 7 (+Ø.625v)
8	$\overline{VF8}$	" " " 8 (+1.25v)
1Ø	$\overline{VF9}$	" " " 9 (+2.5v)
12	$\overline{VF1Ø}$	" " " 1Ø(+5.Øv)
14	$\overline{VF11}$	" " " 11(-1Ø.Øv)

SOCKET J3 - Data Output Port No 6

Pin	Name	Signal Function
1		Not Used
2		" "
3		" "
5		" "
7		" "
8		" "
1Ø		" "
12		" "
14		" "

SOCKET J4 - Data Output Port No 7

Pin	Name	Signal Function
1		Not Used
2		" "
3		" "
5		" "
7		" "
8		" "
1Ø		" "
12		" "
14		" "

SOCKET J5 - Data Input Port No 4

Pin	Name	Signal Function
1		Not Used
2		" "
3		" "
5		" "
7		" "
8		" "
1Ø	$\overline{\text{VTØ}}$	Terminal Voltage Bit Ø (+Ø.25v)
12	$\overline{\text{VT1}}$	" " " 1 (+Ø.5v)
14	$\overline{\text{VT2}}$	" " " 2 (+1.Øv)

SOCKET J6 - Data Input Port No 5

Pin	Name	Signal Function
1	$\overline{\text{VT3}}$	Terminal Voltage Bit 3 (+2.Øv)
2	$\overline{\text{VT DATA TMTD}}$	" " Data Transmitted
3	$\overline{\text{VT4}}$	" " Bit 4 (+4.Øv)
5	$\overline{\text{VT5}}$	" " " 5 (+8.Øv)
7	$\overline{\text{VT6}}$	" " " 6 (+16.Øv)
8	$\overline{\text{VT7}}$	" " " 7 (+32.Øv)
1Ø	$\overline{\text{VT8}}$	" " " 8 (+64.Øv)
12	$\overline{\text{VT9}}$	" " " 9 (+128.Øv)
14	$\overline{\text{VT1Ø}}$	" " " 1Ø (-256.Øv)

SOCKET J7 - Data Input Port No 6

Pin	Name	Signal Function
1		Not Used
2		" "
3		" "
5		" "

7		Not Used
8		" "
1Ø		" "
12	$\overline{EQØ}$	E'_q Voltage Bit Ø (+Ø.ØØ488v)
14	$\overline{EQ1}$	" " " 1 (+Ø.ØØ966v)

SOCKET J8 - Data Input Port No 7

Pin	Name	Signal Function
1	$\overline{EQ2}$	E'_q Voltage Bit 2 (+Ø.Ø1953v)
2	$\overline{EQ}$ DATA TMTD	" " Data Transmitted
3	$\overline{EQ3}$	" " Bit 3 (+Ø.Ø39Ø6v)
5	$\overline{EQ4}$	" " " 4 (+Ø.Ø78l3v)
7	$\overline{EQ5}$	" " " 5 (+Ø.15625v)
8	$\overline{EQ6}$	" " " 6 (+Ø.3125v)
1Ø	$\overline{EQ7}$	" " " 7 (+Ø.625v)
12	$\overline{EQ8}$	" " " 8 (+1.25v)
14	$\overline{EQ9}$	" " " 9 (-2.5v)

SOCKET J9 - Data Input Port No 2Ø

Pin	Name	Signal Function
1	SPBØ	Speed Count No 2 Bit Ø (1uS)
2		Not Used
3	SPBl	Speed Count No 2 Bit 1 (2uS)
5	SPB2	" " " " " 2 (4uS)
7	SPB3	" " " " " 3 (8uS)
8	SPB4	" " " " " 4 (16uS)
1Ø	SPB5	" " " " " 5 (32uS)
12	SPB6	" " " " " 6 (64uS)
14	SPB7	" " " " " 7 (128uS)

SOCKET J1Ø - Data Input Port No 21

Pin	Name	Signal Function
1	SPB8	Speed Count No 2 Bit 8 (Ø.256mS)
2		Not Used
3	SPB9	Speed Count No 2 Bit 9 (Ø.512mS)
5	SPBlØ	" " " " " 1Ø (1.Ø24mS)
7	SPBll	" " " " " 11 (2.Ø48mS)
8	SPBl2	" " " " " 12 (4.Ø96mS)
1Ø	SPBl3	" " " " " 13 (8.192mS)
12	SPBl4	" " " " " 14 (16.384mS)
14	SPBl5	" " " " " 15 (-32.768mS)

SOCKET J11 - Data Input Port No 22

Pin	Name	Signal Function
1	$\overline{\text{DELBØ}}$	Shaft Angle No 2 Bit Ø (1uS)
2		Not Used
3	$\overline{\text{DELB1}}$	Shaft Angle No 2 Bit 1 (2uS)
5	$\overline{\text{DELB2}}$	" " " " 2 (4uS)
7	$\overline{\text{DELB3}}$	" " " " 3 (8uS)
8	$\overline{\text{DELB4}}$	" " " " 4 (16uS)
1Ø	$\overline{\text{DELB5}}$	" " " " 5 (32uS)
12	$\overline{\text{DELB6}}$	" " " " 6 (64uS)
14	DELB7	" " " " 7 (128uS)

SOCKET J12 - Data Input Port No 23

Pin	Name	Signal Function
1	$\overline{\text{DELB8}}$	Shaft Angle No 2 Bit 8 (Ø.256mS)
2	$\overline{\text{DELB DATA TMTD}}$	" " " Data Transmitted
3	$\overline{\text{DELB9}}$	" " " Bit 9 (Ø.512mS)
5	$\overline{\text{DELB1Ø}}$	" " " " 1Ø (1.Ø24mS)
7	$\overline{\text{DELB11}}$	" " " " 11 (2.Ø48mS)
8	$\overline{\text{DELB12}}$	" " " " 12 (4.Ø96mS)
1Ø	$\overline{\text{DELB13}}$	" " " " 13 (8.192mS)
12	$\overline{\text{DELB14}}$	" " " " 14 (16.384mS)
14	DELB15	" " " " 15 (-32.768mS)

EDGE CONNECTOR P1 - Microprocessor Bus Connector

Pin	Name	Signal Function
1		Not Used
2		" "
3	GND	Supply Common
4	GND	" "
5		Not Used
6	$\overline{\text{MM2}}$	Auxilary Address Bit
7	MM2	" " " (Complement)
8		Not Used
9		" "
1Ø		" "
11		" "
12		" "
13		" "
14		" "
15		" "
16		" "
17		" "
18		" "
19	MAD8	Address Bit 8
2Ø	MAD9	" " 9

21		Not Used
22		" "
23		" "
24	DBØ	Output Data Bit Ø
25		Not Used
26	DB1	Output Data Bit 1
27		Not Used
28	DB3	Output Data Bit 3
29		Not Used
3Ø	DB2	Output Data Bit 2
31		Not Used
32	DB5	Output Data Bit 5
33		Not Used
34	DB4	Output Data Bit 4
35		Not Used
36	DB7	Output Data Bit 7
37		Not Used
38	DB6	Output Data Bit 6
39		Not Used
4Ø		" "
41		" "
42		" "
43		" "
44		" "
45		" "
46		" "
47		" "
48		" "
49		" "
5Ø		" "
51		" "
52		" "
53		" "
54	$\overline{\text{I/O OUT}}$	I/O Output Strobe
55		Not Used
56		" "
57	$\overline{\text{DA12}}$	Device Address Bit 12 (Complement)
58		Not Used
59	DA13	Device Address Bit 13
6Ø	DA12	Device " " 12
61		Not Used
62	$\overline{\text{DA13}}$	Device Address Bit 13 (Complement)
63	$\overline{\text{DA14}}$	" " " 14 (")
64	DS14	" Select Bit 14
65	DA15	" Address Bit 15
66	DA14	" " 14
67	$\overline{\text{DS15}}$	" Select Bit 15
68	$\overline{\text{DA15}}$	" Address Bit 15 (Complement)
69		Not Used
7Ø	$\overline{\text{INØ}}$	Peripheral Input Bit Ø
71	$\overline{\text{DA11}}$	Device Address Bit 11 (Complement)
72	IN1	Peripheral Input Bit 1
73	DS11	Device Select Bit 11
74	DA11	" Address Bit 11
75		Not Used
76	IN3	Peripheral Input Bit 3
77		Not Used

78	IN2	Peripheral Input Bit 2
79	IN5	" " " 5
8Ø		Not Used
81	IN6	Peripheral Input Bit 6
82	I/O IN	I/O Input Strobe
83	I/O MOD ENBL	I/O Module Enable
84	IN4	Peripheral Input Bit 4
85	DA1Ø	Device Address Bit 1Ø (Complement)
86	IN7	Peripheral Input Bit 7
87	DS1Ø	Device Select Bit 1Ø
88	DA1Ø	" Address Bit 1Ø
89		Not Used
9Ø		" "
91		" "
92		" "
93		" "
94	MAD11	Address Bit 11
95		Not Used
96	MAD1Ø	Address Bit 1Ø
97		Not Used
98		" "
99	+5VDC	+5 volt Source Power
1ØØ	+5VDC	+5 " " "

6.3.2 Installation Data

Connector to microprocessor bus:-

Dual 5Ø-pin edge connector Ø.125" centres
(Compatible with SAE type C8ØØ1ØØ
or CDC type VPBØ1C5ØEØØA1
or Viking Industries 3VH5Ø / 1CN5)

Connector to Input/Output Ports:-

14-pin DIL socket Ø.1" centres
(Compatible with Jermyn type A23-2Ø48LC)

Power Supply Requirements:-

+5vdc ±5% @ Ø.9 A typ (1.3A max)

Operating Temperature Range:-

Ø°C to +7Ø°C

6.3.3 Components Specification

Integrated Circuits

A1	=	P34Ø4	B7	=	N8234B
A2	=	P34Ø4	B8	=	N8234B
A3	=	P34Ø4	B9	=	N8234B
B1	=	P34Ø4	B1Ø	=	N8234B
B2	=	P34Ø4	B11	=	N8234B
B3	=	P34Ø4	C1	=	P32Ø5
B4	=	N8234B	C2	=	P32Ø5
B5	=	N8234B	C3	=	SN743ØN
B6	=	N8234B	C4	=	P34Ø4

Resistors

R1 to R66	1KΩ, $\frac{1}{8}$ Watt, metal oxide

Capacitors

C1	25µf, 25V, electrolytic
C2 to C14	.Ø1 µf ceramic

Sockets

J1 to J12	A23-2Ø48LC or A23-2Ø28Z

Headers

J24-2148 (12 off)

6.3.4 Component Layout

The component layout is given in Fig 6.4.

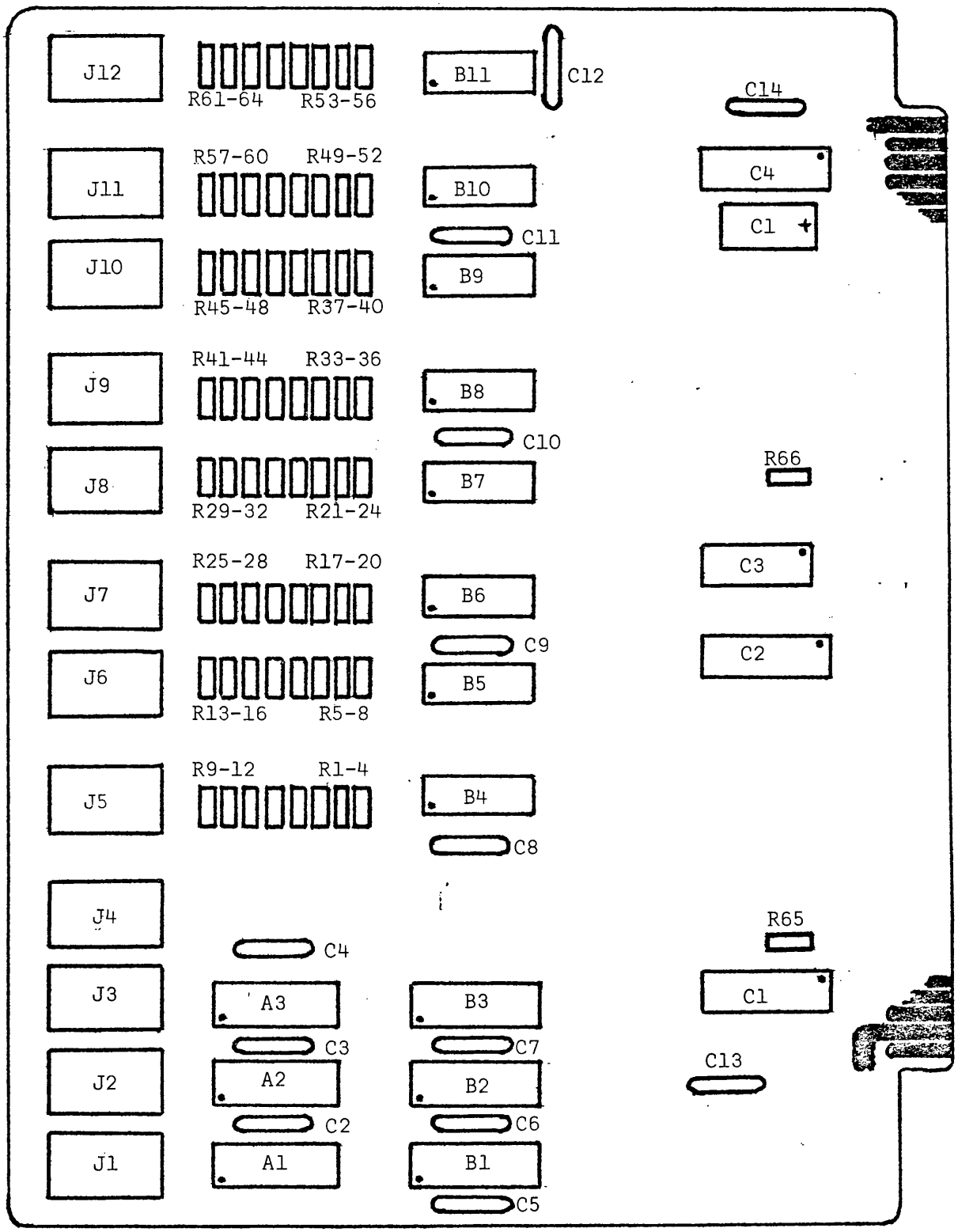


Fig 6.4 Input/Output Card Component Layout (view from above)

SECTION 7

PERIPHERAL INTERRUPT PRIORITY CARD

The application of multi-level interrupts to the I8Ø8Ø processor has been discussed in Section 2.3.3. The function of the Peripheral Interrupt Priority Card is to handle the granting of interrupts to the microprocessor system peripherals in a controlled sequence of priorities (the PDP-11/2Ø interrupts directly through the Interface Card on the highest priority). The Peripheral Interrupt Card has been designed to be general in application so that it may be used with any system of peripherals and is 'stackable' such that the number of interrupts which may be handled can be readily extended.

7.1 GENERAL FUNCTIONAL DESCRIPTION

The Peripheral Interrupt Priority Card functions on the 'daisy-chain' principle for servicing the requests of the interrupting peripherals as shown in Fig 7.1. The interrupt requests from the various peripheral devices may occur at any instant in time once each peripheral has been "interrupt enabled" (Sections 8 to 11) and started by the microprocessor or other device (eg DMA by PDP-11 of relevant control port). These requests, numbered 2 to 7 in Fig 7.1, set the request flip-flops whose outputs pass through the Interrupt Request Latch Block onto the common Interrupt Request bus. This request is passed via the Interface Card to the microprocessor, which will respond to the request if not disabled or not in a higher priority interrupt mode (ie PDP-11 transfer via the Interface Card) by sending an Interrupt Acknowledge signal.

This acknowledgement will cause the whole Interrupt Request Latch Block to be 'frozen' such that no changes may be made to the state of the requests in this block. However, during this period, further requests may come in to set the individual flip-flops but these will not cause any change in the 'frozen' block. The Interrupt Acknowledge signal then ripples through the Latch Block

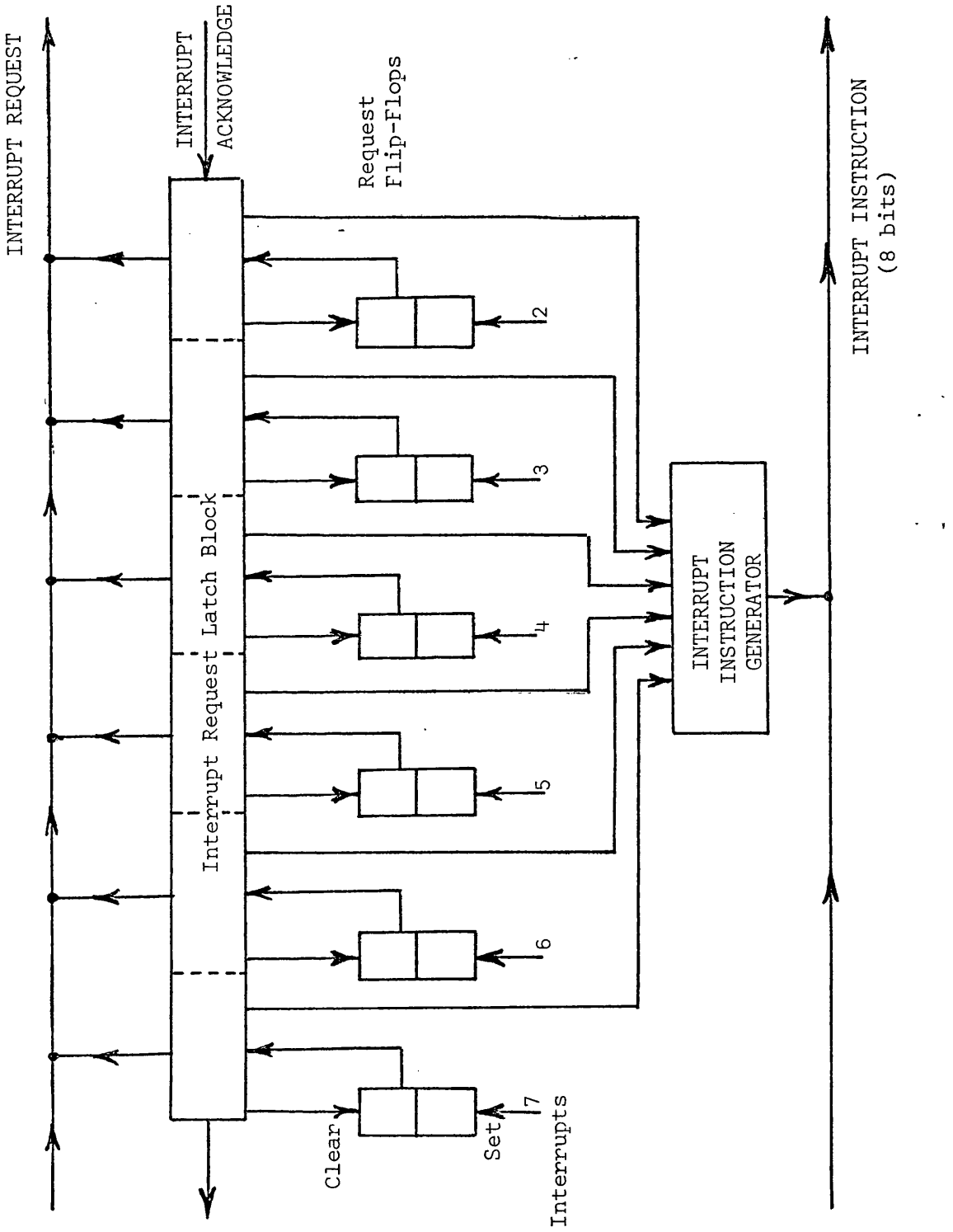


Fig 7.1 Peripheral Interrupt Priority Card-Functional Block Diagram

until it comes to the first stage which has an interrupt request set, and then passes no further. An appropriate signal is sent, from this stage to the Interrupt Instruction Generator which generates a specific interrupt instruction for the stage which has been acknowledged. This causes an Interrupt Cycle to be entered as explained in the reference manual[67]. The interrupt flip-flop of this stage is reset at this time.

At the end of the interrupt instruction cycle, the Interrupt Acknowledge signal is removed which 'unfreezes' the Latch Block. This allows any new or outstanding interrupt requests to cause the cycle to be repeated.

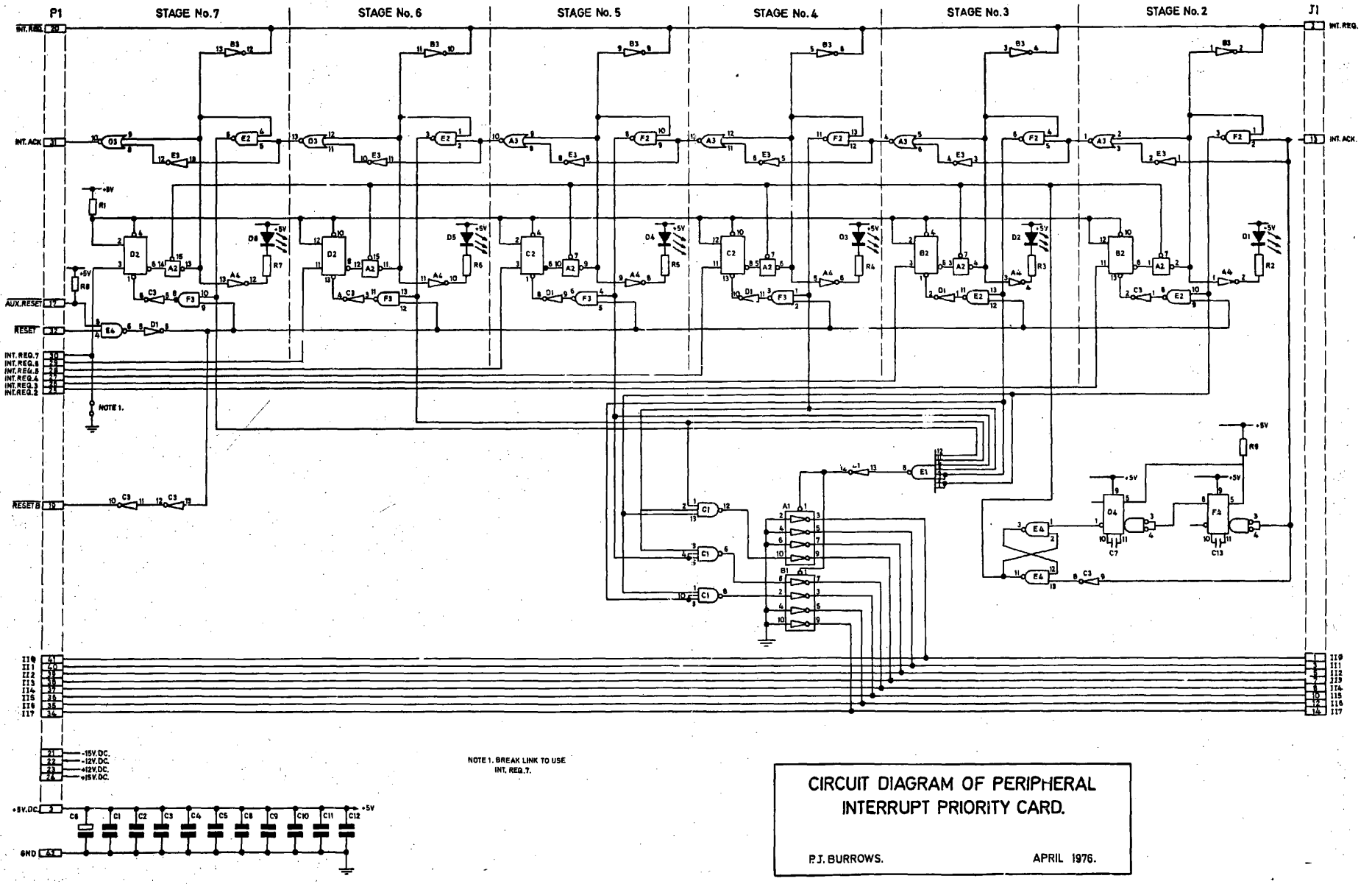
7.2 THEORY OF OPERATION

The circuit diagram of the Peripheral Interrupt Priority Card is given in Fig 7.2. An interrupt sequence is initiated by the appearance of a positive rising edge at one of the six INT REQ inputs (P1, pins 25 to 3Ø). Thus the INT REQ signal may be a pulse of logical 'Ø'or logical '1' or a level change from 'Ø' to '1'. This request causes one of the bistable latches in B2, C2 or D2 to be set such that the '$\bar{Q}$' output (pin 6 or 8) goes to a logical 'Ø' level. In the following description, it will be assumed that such pulses have arrived simultaneously at the INT REQ 3 and INT REQ 5 inputs, thus causing B2, pin 6 and C2, pin 6 to be 'Ø'.

The signals from B2 and C2 pins 6 pass through their respective sections in the Latch Block A2, causing pins 4 and 9 of A2 to be '1'. In turn, these signals pass through the open-collector buffer stage, B3, causing pins 4 and 8 to be a logical 'Ø' level. The outputs from the buffer stage, B3, are commoned together onto the $\overline{\text{INT REQ}}$ bus, which is pulled low by both outputs, passing an interrupt request to the Interface Card via J1, pin 2. Assuming the Interface Card and microprocessor are able to respond (Section 2.3.3), an acknowledgement is received in the form of the INT ACK line (J1, pin 13) going to a logical '1' level.

This INT ACK signal causes C3, pin 8 to go low which causes E4, pin 11 to be high regardless of the input to E4, pin 12. As the

Fig. 7.2 Peripheral Interrupt Priority Card Circuit Diagram



NOTE 1. BREAK LINK TO USE INT. REQ.7.

CIRCUIT DIAGRAM OF PERIPHERAL INTERRUPT PRIORITY CARD.

P.J. BURROWS. APRIL 1976.

output of E4 (pin 11) is connected to the control pins 7 and 15 of the Latch Block, A2, the latter is immediately latched, preventing any further change in state of its outputs. From this point in time, any further INT REQ signals may set the latches in B2, C2 and D2 but will not be active in this immediate interrupt cycle.

Meanwhile, the INT ACK signal on J1, pin 13 is proceeding to ripple through the daisy chain consisting of A3, E3 and F2 and parts of E2 and D3. At the first stage (No 2), pin 1 of F2 will be a logical 'Ø' as there was no request from the stage and gate F2 is closed producing no action in stage No 2. However, the INT ACK output gate from this stage (section 1, 2 and 3 of A3) will be open due to the presence of the logical 'Ø' on pin 2 and so the INT ACK signal will be passed via A3, pin 1 to the next stage (No 3).

As stage No 3 has requested an interrupt, pin 4 of F2 will be a logical '1' and thus the associated gate is open, passing a logical 'Ø' to F2, pin 6. Meanwhile, pin 5 of A3 is also a logical '1', which causes the gate output at A3, pin 4 to be logical 'Ø' regardless of the state of the other input and, thus, the INT ACK signal is prevented from passing further down the daisy chain once it has reached this stage with an INT REQ set. The 'low' signal from F2, pin 6 performs two functions: it resets the flip-flop in B2 via the 11, 12, 13 section of E2 and the 1, 2 section of D1; it also acts as a command signal to the Interrupt Instruction Generator section. The Instruction Generator comprises elements A1, B1, C1, E1 and the 12, 13 section of D1. The logical 'Ø' from F2, pin 6 appears on pins 3 and 4 of E1 which causes the output, pin 8, to go high thus enabling the tri-state buffers A1 and B1. The outputs of these buffers are connected to the Interrupt Instruction bus which is now driven by this module, whereas it has previously been 'floating'.

The logical 'Ø' from F2, pin 6 is also applied to pins 9 and 1Ø of C1 which causes the output, pin 8, to go to a logical '1'. Pins 6 and 12 of C1 are still a logical 'Ø' and thus the inputs to B1 (pins 1Ø, 4, 6 and 2) and A1 (pins 1Ø, 6, 4 and 2) are 'Ø','Ø', '1',

'Ø', 'Ø', 'Ø', 'Ø' and 'Ø' respectively. As these tri-state buffers logically complement the data, the instruction appearing on II7 to IIØ is '11Ø11111'. In the 8Ø8Ø instruction set, this is an 'RST 3Q' instruction which causes an interrupt subroutine call to location $3Ø_8$. Subsequent program flow is dependent on the instructions of the interrupt subroutine.

Once the interrupt instruction has been read by the microprocessor, the INT ACK on pin 13 of J1 is removed. This causes the daisy chain to relax to its original 'off' condition. However, it is important that no interrupts are passed through it while it is rippling through to its settled state, as these would cause spurious $\overline{\text{INT REQ}}$ signals to be generated. Therefore, the Latch Block A2 is held latched for approximately 600nS while this settling is taking place. This action is performed by the timing monostables D4 and F4 whose function is to ensure that E4, pin 11 stays high for 600nS after J1, pin 13 goes low, thus holding pins 7 and 15 of A2 for this period.

In the first part of this description, it was assumed that INT REQ 3 and INT REQ 5 inputs had been received simultaneously and thus the $\overline{\text{INT REQ}}$ line will still be held low by the signal from stage No 5. The above process will now be repeated in a similar fashion to acknowledge this interrupt as soon as the CPU is once more 'interrupt enabled'. However, if any higher priority interrupt requests have occurred before this second cycle is entered the acknowledgement to stage No 5 will be deferred until they have been processed.

A facility for resetting (clearing) all the interrupt requests is provided in two forms. A signal is provided from the general $\overline{\text{RESET}}$ bus (sections 2.2.5 and 5.1.1) which causes all the interrupt request flip-flops to be cleared by a RESET operation. This appears at P1, pin 32 and passes through the negative logic OR gate in the 4, 5, 6 section of E4 to the flip-flops and is also buffered and chained to other cards as the $\overline{\text{RESET B}}$ signal at pin 19 of P1. An $\overline{\text{AUX RESET}}$ input is also provided at pin 17 of P1 to allow the requests to be cleared by the microprocessor under program control without the program-resetting implications of a general RESET. This

signal is derived from Output Port No 1, bit 7 (see Section 6.3.1). Due to the OR gate in E4, a $\overline{\text{RESET B}}$ pulse is generated by either of these signals.

For debugging and error-detection purposes, a set of display LED s are provided, D1 to D6. The diodes show the existence of any unacknowledged interrupt requests. A single interrupt request or the highest priority request of a group will not cause a LED display to light as the INT ACK signal is generated by a hardware latch on the CPU and regardless of whether the CPU is interrupt enabled or not, and thus the first request of any group will always be acknowledged.

7.3 UTILIZATION

7.3.1 Connections List

SOCKET J1 - Interrupt Instruction Output

Pin	Name	Signal Function
1	$\overline{\text{II}\emptyset}$	Interrupt Instruction Bit Ø
2	$\overline{\text{INT REQ}}$	Interrupt Request to Interface
3	II1	Interrupt Instruction Bit 1
4		Not Used
5	II2	Interrupt Instruction Bit 2
6		Not Used
7	II3	Interrupt Instruction Bit 3
8	II4	" " Bit 4
9		Not Used
1Ø	II5	Interrupt Instruction Bit 5
11		Not Used
12	II6	Interrupt Instruction Bit 6
13	INT ACK	Interrupt Acknowledge from Interface
14	II7	Interrupt Instruction Bit 7

EDGE CONNECTOR P1 - Peripheral Bus Interconnector

Pin	Name	Signal Function
1		Not Used
2		" "
3	GND	Supply Common
4		Not Used
5		" "
6		" "

7	XX	Reference Slot
8		Not Used
9		" "
1Ø		" "
11		" "
12		" "
13		" "
14		" "
15		" "
16		" "
17	$\overline{\text{AUX RESET}}$	Auxilary Reset Signal
18		Not Used
19	$\overline{\text{RESET B}}$	Buffered Peripheral Reset Signal
2Ø	$\overline{\text{INT REQ}}$	Peripheral Interrupt Request
21	-15 VDC	-15 volt Source Power
22	-12 VDC	-12 volt Source Power
23	+12 VDC	+12 volt Source Power
24	+15 VDC	+15 volt Source Power
25	INT REQ 2	Peripheral Interrupt Request No 2 ('RST 2Q')
26	INT REQ 3	" " " No 3 ('RST 3Q')
27	INT REQ 4	" " " No 4 ('RST 4Q')
28	INT REQ 5	" " " No 5 ('RST 5Q')
29	INT REQ 6	" " " No 6 ('RST 6Q')
3Ø	INT REQ 7	" " " No 7 ('RST 7Q')
31	$\overline{\text{INT ACK}}$	Interrupt Acknowledge to lower priority inter-
32	$\overline{\text{RESET}}$	Reset signal from microprocessor bus rupts
33		Not Used
34	II7	Interrupt Instruction Bit 7
35	II6	" " Bit 6
36	II5	" " Bit 5
37	II4	" " Bit 4
38	II3	" " Bit 3
39	II2	" " Bit 2
4Ø	II1	" " Bit 1
41	IIØ	" " Bit Ø
42	+5 VDC	+5 volt Source Power
43		Not Used

7.3.2 Installation Data

Connector to Peripheral Unit Bus:

Single 4Ø-pin PC Edge Connector. Ø.1" centres
(Compatible with Vero type 13623-1)

Power Supply Requirements:

+5 vdc ±5% @ 500mA typ (800mA max)

Operating Temperature Range:

$+Ø^{\circ}C$ to $+7Ø^{\circ}C$

7.3.3 Components Specification

Integrated Circuits

A1	=	SN74368N	D2	=	SN7474N
A2	=	P34Ø4	D3	=	SN74Ø2N
A3	=	SN74Ø2N	D4	=	SN74121N
A4	=	SN74Ø4N	E1	=	SN743ØN
B1	=	SN74368N	E2	=	SN74ØØN
B2	=	SN7474N	E3	=	SN74Ø4N
B3	=	SN74Ø5N	E4	=	SN74ØØN
C1	=	SN741ØN	F2	=	SN74ØØN
C2	=	SN7474N	F3	=	SN74ØØN
C3	=	SN74Ø4N	F4	=	SN74121N
D1	=	SN74Ø4N			

Resistors

R1	1 KΩ, $\frac{1}{8}$ W, metal oxide
R2 to R7	270 Ω, $\frac{1}{8}$ W, metal oxide
R8, R9	1 KΩ, $\frac{1}{8}$ W, metal oxide

Capacitors

C1	0.1 µf, ceramic
C2 to C5	.01 µf, ceramic
C6	47 µf, 25 V, electrolytic
C7	470 pf
C8 to C11	.01 µf ceramic
C12	0.1 µf ceramic
C13	470 pf

Diodes

D1 to D6	LED (Red)

Socket

J1	A23-2048LC

SECTION 8

LOAD ANGLE AND SPEED TRANSDUCER UNIT

The Load Angle and Speed Transducer Unit has been designed to provide direct digital readings relating to the micromachine shaft position and speed, thus enabling the microprocessor to calculate the load angle and rotor transient velocity. The complete unit comprises four modules: one Zero-Crossing Detector Card, one (two-channel) Photo-cell Amplifier Module and two Shaft Angle and Speed Counter Cards.

8.1 GENERAL FUNCTIONAL DESCRIPTION

The micromachine model is a four-pole machine in which a full cycle of 360 electrical degrees of the terminal voltage nominally corresponds to a shaft rotation of 180 degrees. Thus, the Load Angle and Speed Transducer Unit basically comprises two duplicate channels which function in the measurement mode during alternate half-cycles of the shaft revolution. During the half-cycle in which each transducer is not in the measurement mode, it transfers the data collected to the microprocessor and is reset for the next reading.

Referring to the functional block diagram of the unit in Fig 8.1, one channel functions in the following manner. The passage of the slot in the shaft-mounted disc past the photo-pickup No 1 causes a START pulse to be sent to the Shaft Angle and Speed Counter Card No 1 via the Photo-cell Amplifier module. This causes the clock pulses from the 2.0 MHz crystal-controlled clock to be counted in the speed counter starting from its zero condition. This count is also made in the shaft angle register which is enabled at this time to copy the speed counter. At the occurrence of a zero-crossing in the phase A voltage of the infinite busbar, the Zero-Crossing Detector Card sends a signal to the angle register which causes it to HOLD its reading, thus preserving a count which is a function of the phase difference

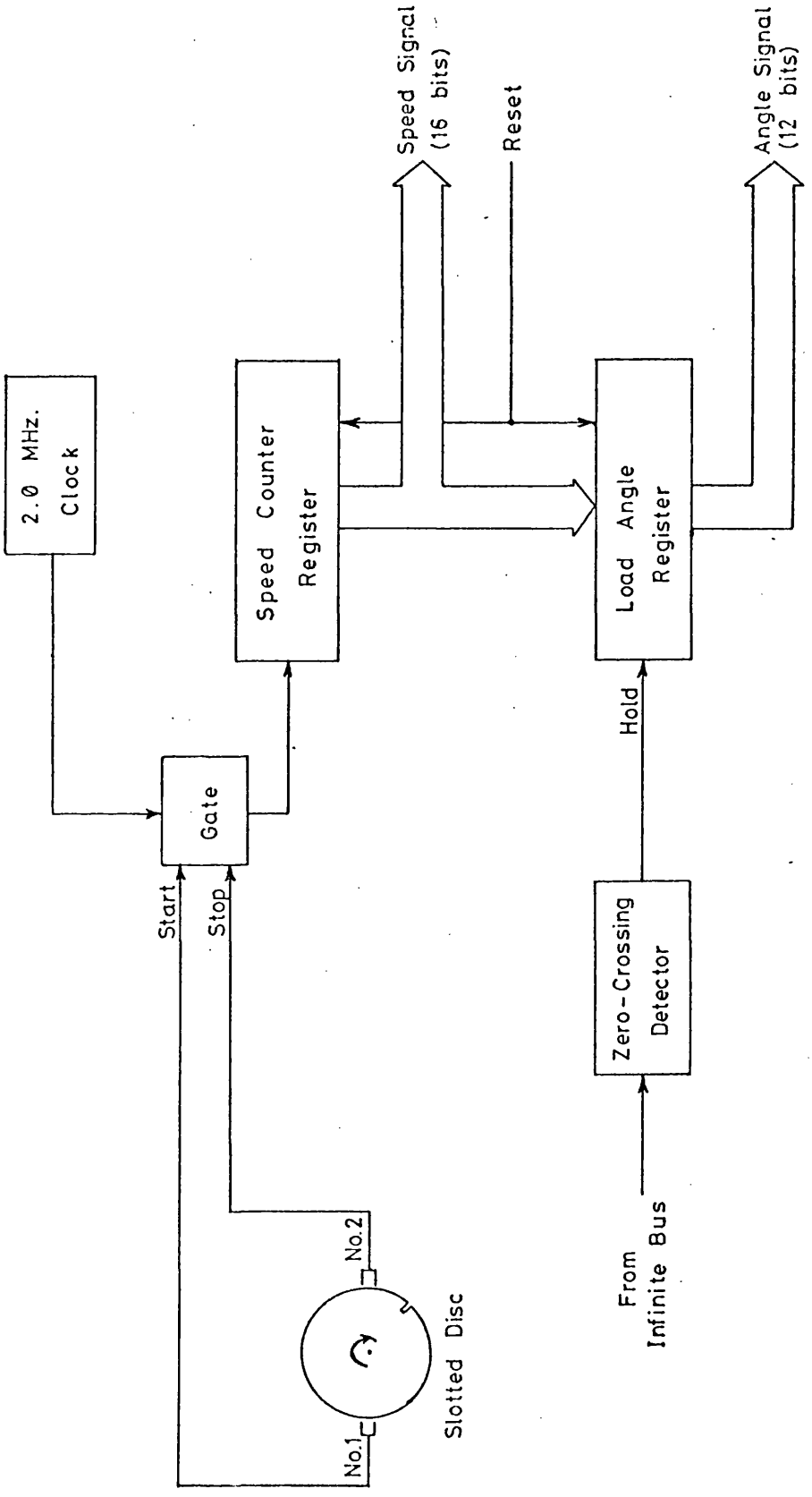


Fig 8.1 Load Angle and Speed Transducer Block Schematic

(one channel, simplified)

between the shaft position and a synchronously rotating reference (the infinite busbar voltage). Although the angle register is now held, the speed counter continues its count until the slot in the disc passes the second photo-pickup. This event causes the counter to stop and a signal is sent to the microprocessor to signify completion of the measurement mode. In normal operation, the transfer of data to the microprocessor and the resetting of the counter occurs before the slot again passes photo-pickup No 1. The system incorporates various 'lock-out' techniques to avoid spurious readings should this transfer not have been completed in the required interval as explained in Section 8.3.

Shaft Angle and Speed Counter No 2 functions in a similar manner except that the measurement mode is initiated by the passage of the slot past photo-pickup No 2 and completion determined by No 1.

8.2 CONTROL AND STATUS REGISTERS

For system flexibility and efficiency of operation, the Load Angle and Speed Transducer Unit can be run in several different program-selectable modes. These modes are selected by the control bits set in Peripheral Control Register No 2 (Output Port No 2 - see Section 6.3.1) and are described below.

Bit	Name	Function
Ø	DELTA1 STRT	START Load Angle and Speed Transducer Unit No 1. Any measurement in progress is aborted and the count will be restarted from zero on the next signal from pickup No 1
2	DELTA2 STRT	START Load Angle and Speed Transducer Unit No 2. Any measurement in progress is aborted and the count will be restarted from zero on the

		next signal from pickup No 2.
3	DELTA INT ENBL	Enable the unit to request an interrupt on completion of the measurement when the DONE flag is set (see below). Card No 1 interrupts to vector $3Ø_8$ ('RST 3Q') and No 2 interrupts to vector $4Ø_8$ ('RST 4Q').
4	DELTA AUTO	Set the unit to AUTO mode. A START signal is generated automatically within the unit when the data has been transferred to the microprocessor. In this mode the input ports must be read in the sequence 16, 17, 18, 19 (Card No 1) and 2Ø, 21, 22, 23 (Card No 2) as the auto-START signal is triggered by reading ports 19 and 23.

The status of the Load Angle and Speed Counter Unit is displayed in Peripheral Status Register No 1 (Input Port No 1) as follows:-

Bit	Name	Function
2	DELTA1 DONE	Measurement DONE by Card No 1 and available for input to the microprocessor. Also acts as a ready signal for initiation of measurement. Read only. Cleared by DELTA1 STRT and RESET signals.
3	DELTA2 DONE	Measurement DONE by Card No 2 and available for input to the microprocessor. Also acts as a ready signal for initiation of measurement.

Read only. Cleared by DELTA2 STRT and RESET signals.

8.3 CIRCUIT DESCRIPTION

8.3.1 Photo-cell Amplifier Module

The circuit diagram of one channel of the Photo-cell Amplifier Module is given in Fig 8.2. The passage of the slot between the two elements of the TIL 138 photo-detector causes the SN74121 monostable circuit to be triggered generating an output pulse of 1.5 microseconds duration. This output pulse is transmitted by the 2N2904A driver stage via a screened co-axial cable of 75Ω impedance to the counter cards.

8.3.2 Zero-Crossing Detector Card

The Zero-Crossing Detector circuit is centred around a type 709 operational amplifier as shown in Fig 8.3. The amplifier is connected in a high-gain positive feedback mode such that it switches rapidly from positive to negative saturation of the output for small input signals. This effectively 'squares up' the sinusoidal input signal, output transition occurring at the zero-crossing points of the input waveform. The output voltage excursions are limited to +4.7v and -0.7v by the zener diode to make them TTL compatible.

8.3.3 Shaft Angle and Speed Counter Card

The circuit diagram of the Shaft Angle and Speed Counter Card is given in Fig 8.4. Measurement operation is started by one of three signals. In any mode of operation the measurement cycle is initiated by the occurrence of a $\overline{\text{DELTA STRT}}$ or a $\overline{\text{RESET}}$ signal at the edge connector P1 (pin 34 or 17). Either of these signals going to a logic level 'low' for a short period (typically 1.5 microseconds) will cause B4, pin 11 to go 'high' applying a 'high' level to A2, pin 6. This latter gate is a

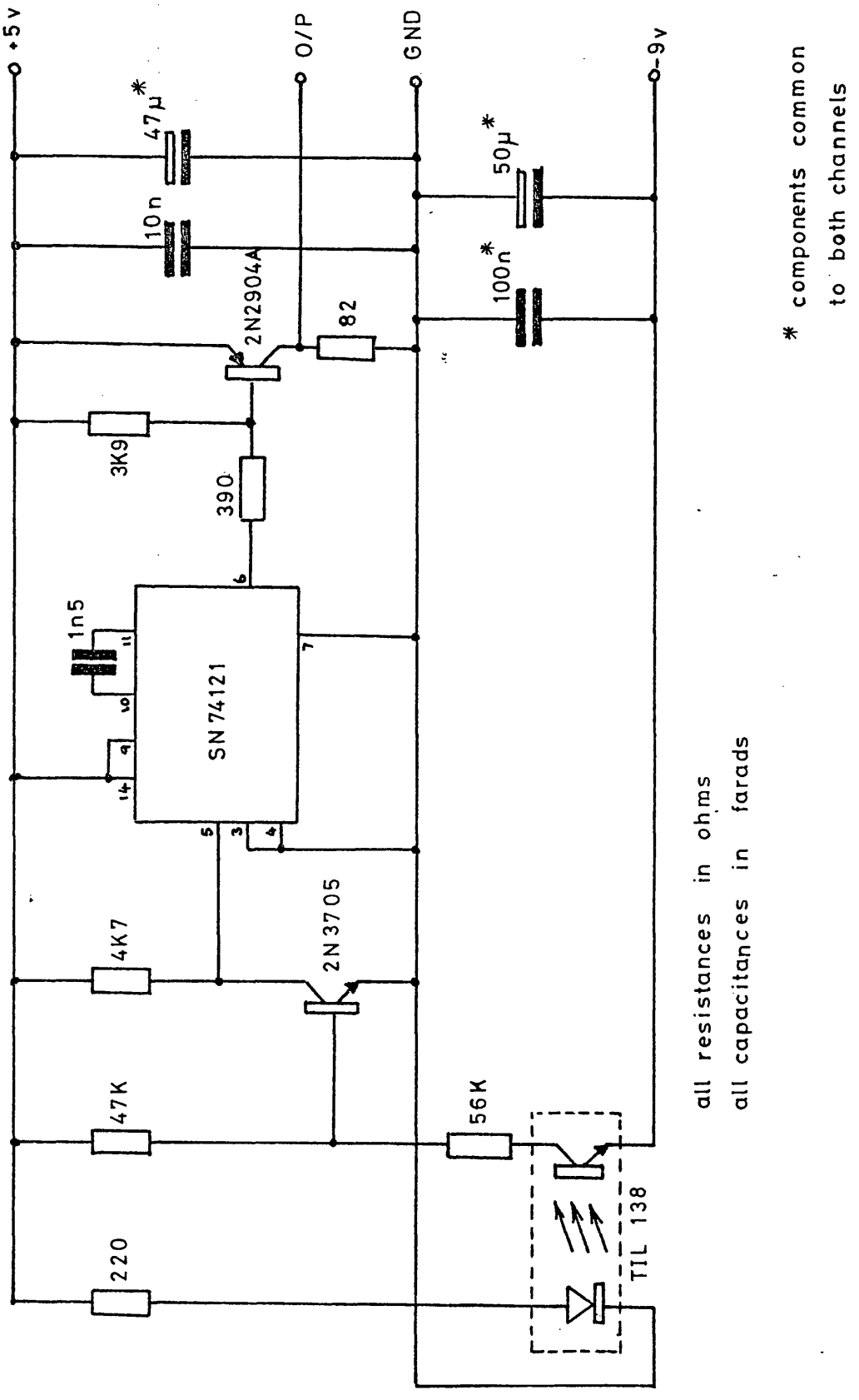


Fig. 8.2 Photo-cell Amplifier Module Circuit Diagram

(one channel)

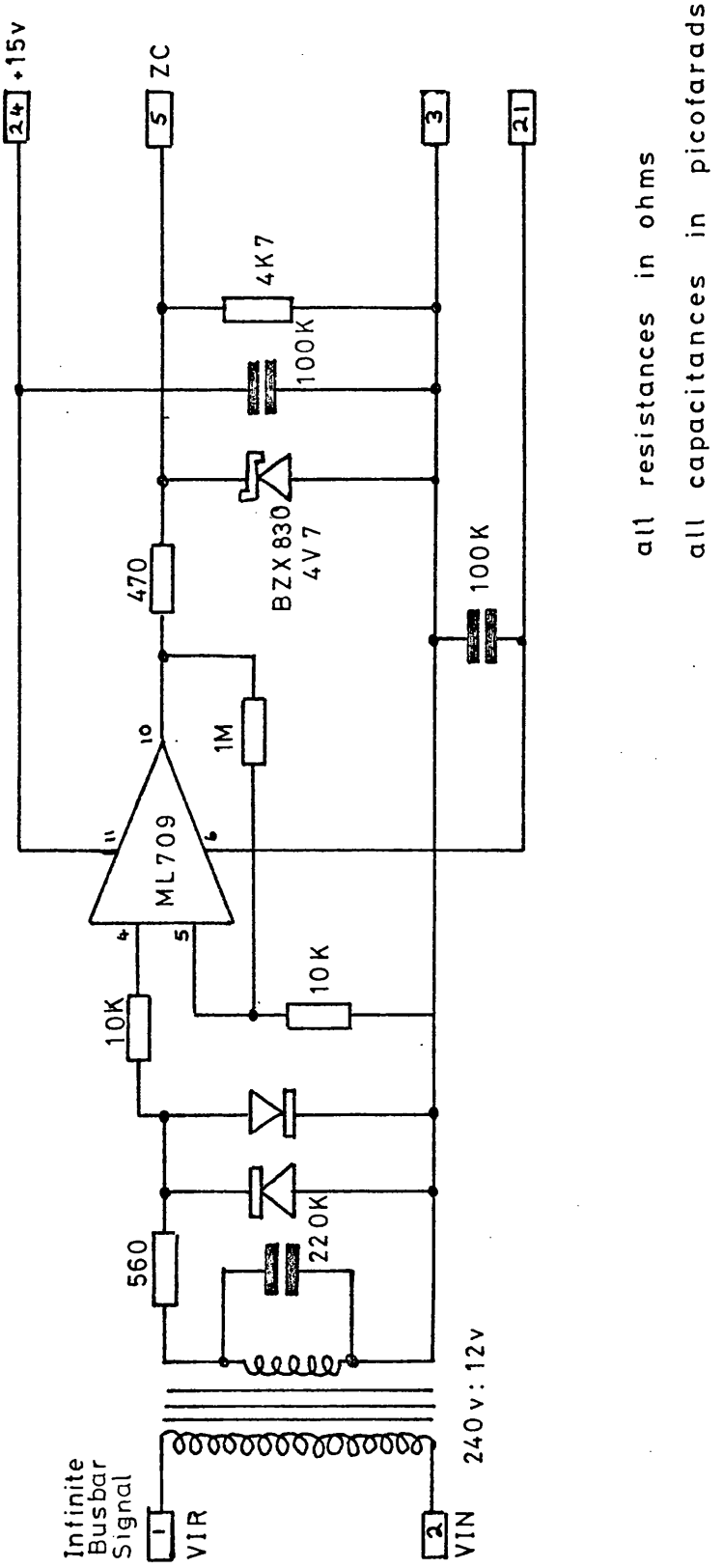
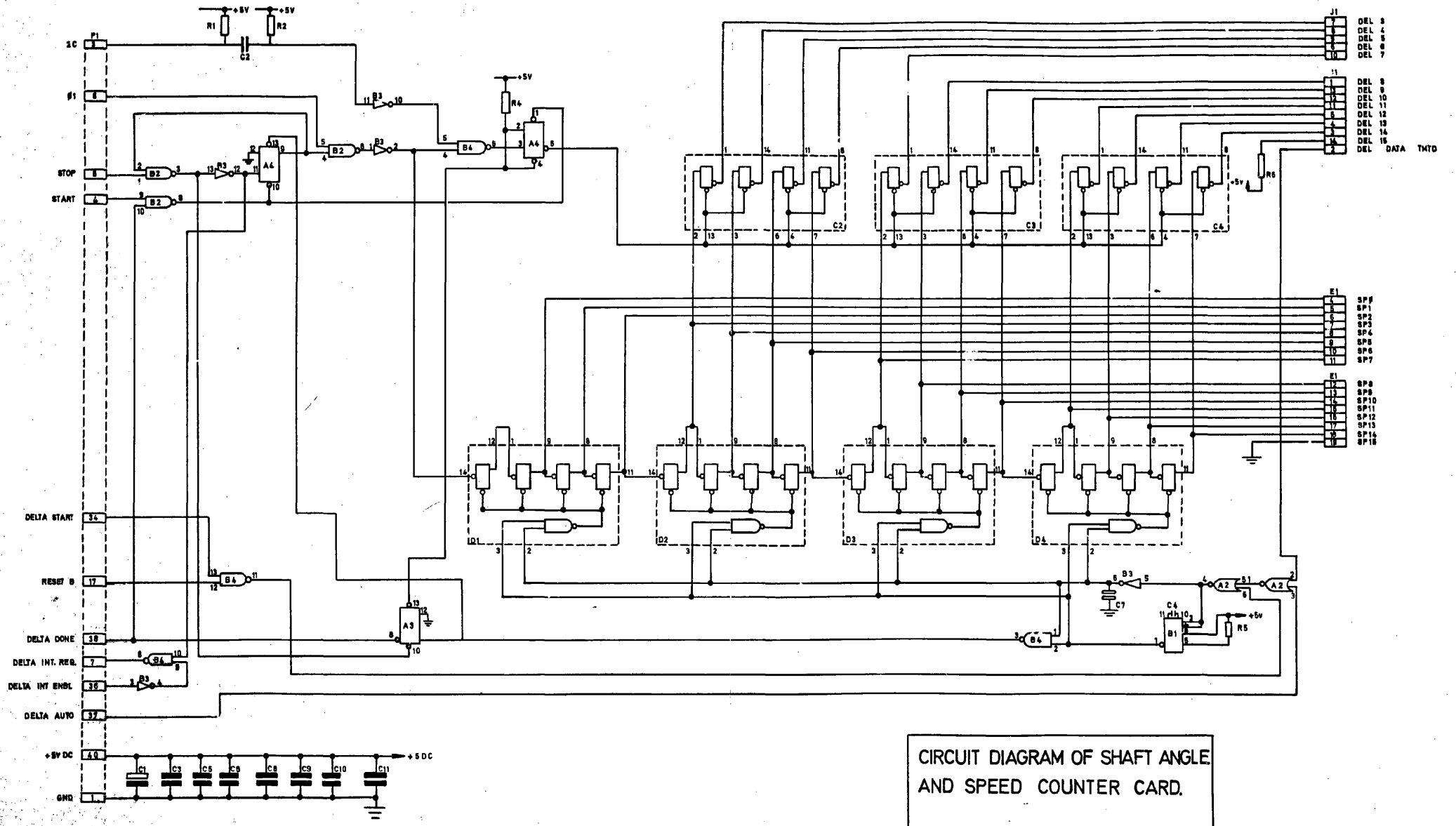


Fig. 8.3 Zero Crossing Detector Circuit Diagram

Fig. 8.4 Load Angle and Speed Transducer Card Circuit Diagram



CIRCUIT DIAGRAM OF SHAFT ANGLE AND SPEED COUNTER CARD.

P.J. BURROWS APRIL 1976

NOR gate which has another input providing for the third starting signal. This starting signal comes from the DATA TMTD pulse on J1, pin 2 which occurs when the microprocessor reads the data on Input Port 19 or 23 (for cards 1 and 2 respectively). This signal is routed via the 1, 2, 3 section of A2 such that it is only passed to A2, pin 5 if the control signal $\overline{\text{DELTA AUTO}}$ (P1, pin 37) is 'low'. This latter condition signifies the AUTO mode of operation. The duration of the pulse from A2, pin 4 is timed by the monostable B1 and its associated circuitry to reject spurious pulses caused by input operations from other ports in the system (Section 6.2.2). Coincident pulses of the correct duration appear on pin 2 and 3 of D1 to D4, the counter stages, which cause a reset to zero count in preparation for the next cycle. Also, the 'low' output pulse on B4, pin 3 causes one latch in each of A3 and A4 to be set such that A3, pin 8 is 'high' and A4, pin 9 is 'low'.

This 'high' level on A3, pin 8 is returned to the microprocessor via P1, pin 38, as the $\overline{\text{DELTA DONE}}$ signal to notify the microprocessor that a measurement reading is in operation. It also causes B2, pin 1Ø to be 'high' opening its gate such that the next START pulse on P1, pin 4 from the first photo-cell amplifier will be passed to the succeeding circuitry. The 'low' level on A4, pin 9 also serves two functions, it is applied to B2, pin 4 to close that gate preventing any $\emptyset_1$ clock pulses (P1, pin 6) from reaching the counter stage and it is also applied to B2, pin 2 to prevent any premature STOP signals from the second photo-cell pickup triggering the completion circuitry. In this manner the circuit state is held awaiting only a START signal from the first photo-cell pickup.

The START pulse from the first photo-cell amplifier causes both latches of A4 to change state as it is applied to A4 pins 1 and 1Ø. This causes A4, pin 9 and A4, pin 6 to go 'high' as shown in Fig 8.5. The 'high' level on A4, pin 9 opens the 4, 5, 6 section gate of B2 allowing the $\emptyset_1$ pulses to pass through to the counter chain D1 to D4. It also causes the 1, 2, 3 section of B2 to be opened so that the circuit is primed to receive the STOP pulse on P1, pin 5. The high level on A4, pin 6 causes

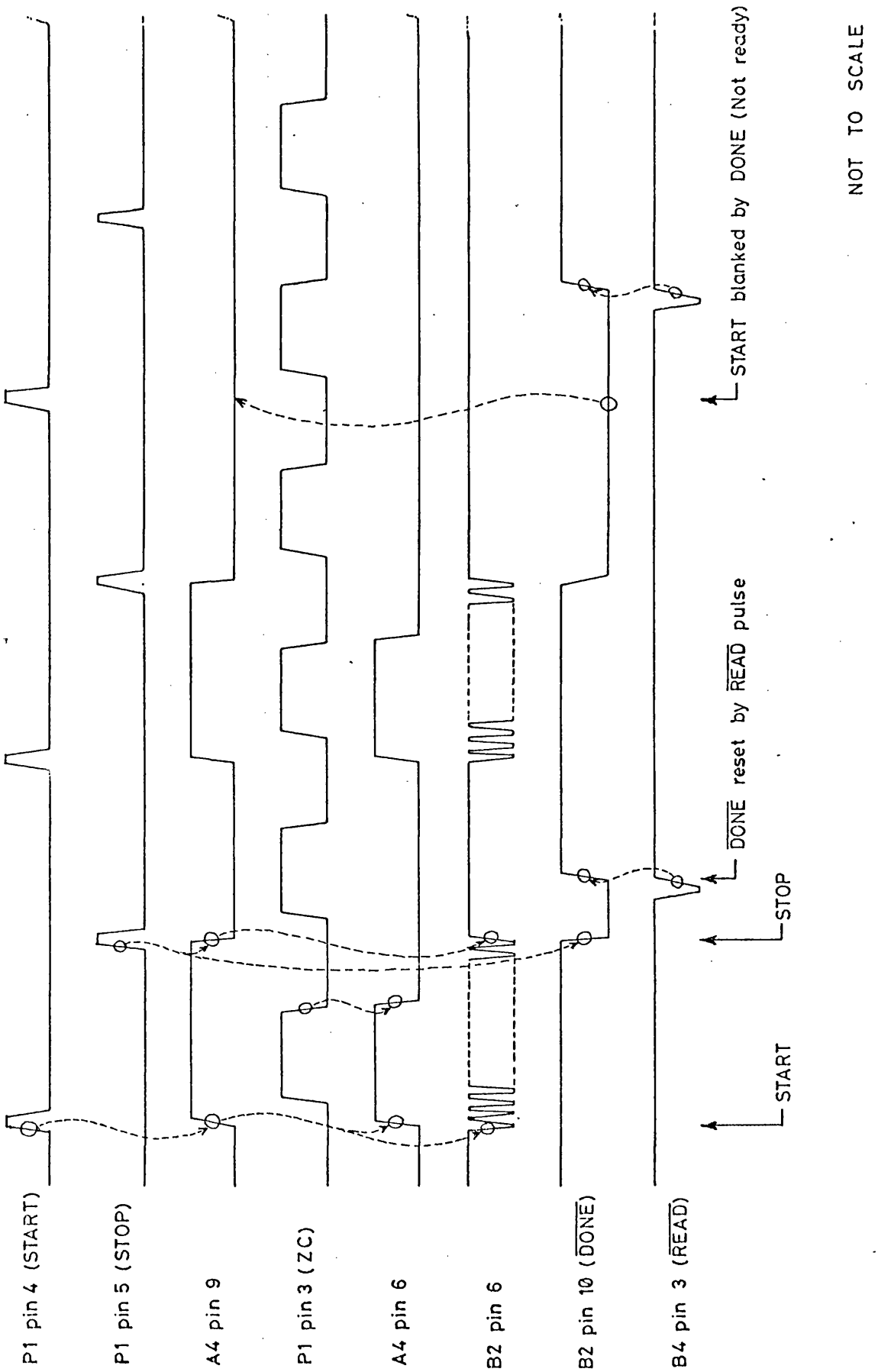


Fig. 8.5 Shaft Angle and Speed Card Timing Diagram (Simplified)

the bistable latch section, C2 to C4, to 'follow' the counter high order sections, recording a count which is a quantised version (in units of eight) of that in D1 to D4.

In normal operation, the next event to occur is the arrival of a pulse from the zero-crossing detector which appears on B4, pin 5 as a 'high' level signal. This signal opens the 4,5,6 section of B4 to pass a signal to A4, pin 3 from the next $\emptyset_1$ clock pulse. The phasing of this signal is arranged to be such that the output of A4 (pin 6) will change state within one clock cycle (Ø.5 µS) but will not occur at a point at which the counter chain D1 to D4 is rippling through. If the latter event did occur, a spurious count would be recorded in the latch section, C2 to C4. This section is now latched with a count which is a function of the shaft position and the counter continues to count until the arrival of a STOP pulse from the second photo-pickup at P1, pin 5. This pulse appears at pin 3 of B2 as a logical 'Ø' pulse which causes one latch in A3 to change state generating a logical 'Ø' level at A3, pin 8. This is transmitted to the microprocessor as the $\overline{\text{DELTA DONE}}$ signal to notify it of the completion of the reading and also closes the 8,9, 1Ø section of B2 locking out any further START pulses.

The pulse from B2, pin 3 is inverted in the 12,13 section of B3 and performs two functions. It causes pin 9 of A4 to be latched 'low' closing off the 4,5,6 section of B2 preventing any more $\emptyset_1$ clock pulses from reaching the counter chain and closing the 1,2,3 section of B2 preventing any subsequent STOP pulses from having any effect on the circuit. It also generates a logical '1' pulse at pin 8 of B4 if P1, pin 36 ($\overline{\text{DELTA INT ENBL}}$) is 'low'. This pulse is transmitted to the Peripheral Interrupt Priority Card as an interrupt request (Section 7). Thus, this interrupt request will only be generated if the microprocessor has previously set a $\overline{\text{DELTA INT ENBL}}$ signal.

If the processor has not set the interrupt mode of operation then it may detect completion by examination of the $\overline{\text{DELTA DONE}}$ flag. In either case it may now read the contents of the latch

sections and counter sections via the appropriate input ports to obtain the information relating to shaft angle and speed. If the AUTO mode of operation has been set this causes the cycle to be repeated as previously explained.

8.4 UTILIZATION

8.4.1 Input and Output Connections

All inputs and outputs of the Load Angle and Speed Transducer Unit are TTL compatible, however, the outputs from the Photo-pickup Amplifier Module have a nominal 75 ohm output impedance to drive the co-axial cables (approximately 4 metres long) connecting it to the counter card.

8.4.1.1 Photo-cell Amplifier Module

The module is mounted in close proximity to the slotted disk on the machine shaft and the input/output connections are hard-wired.

Pin	Name	Signal Function
	+5VDC	+5 volt Source Power
	SA1	Pulse output of Channel 1
	SA2	Pulse output of Channel 2
	GND	Supply Common
	-9VDC	-9 volt Source Power

8.4.1.2 Zero-Crossing Detector Card

EDGE CONNECTOR P1 - Peripheral Bus Interconnector

Pin	Name	Signal Function
1	VIR	Infinite Bus 'Red' Phase Signal
2	VIN	Infinite Bus Neutral
3	GND	Supply Common
4		Not Used
5	ZC	Infinite Bus Zero-Crossing Signal
6		Not Used
7	XX	Reference Slot
8		Not Used

9		Not Used
1Ø		" "
11		" "
12		" "
13		" "
14		" "
15		" "
16		" "
17		" "
18		" "
19		" "
2Ø		" "
21	-15VDC	-15 volt Source Power
22	-12VDC	-12 volt " "
23	+12VDC	+12 volt " "
24	+15VDC	+15 volt " "
25		Not Used
26		" "
27		" "
28		" "
29		" "
3Ø		" "
31		" "
32		" "
33		" "
34		" "
35		" "
36		" "
37		" "
38		" "
39		" "
4Ø		" "
41		" "
42	+5VDC	+5 volt Source Power
43		Not Used

8.4.1.3 Shaft Angle and Speed Counter Card No 1

SOCKET J1 - Load Angle Output Port

Pin	Name	Signal Function
1	DELA 8	Shaft Angle Bit 8
2	DELA DATA TMTD	" " Data Transmitted
3	DELA 14	" " Bit 14
4	DELA 13	" " " 13
5	DELA 12	" " " 12
6	DELA 6	" " " 6
7	DELA 3	" " " 3
8	DELA 4	" " " 4
9	DELA 5	" " " 5
1Ø	DELA 7	" " " 7
11	DELA 11	" " " 11
12	DELA 1Ø	" " " 1Ø

13	$\overline{\text{DELA 9}}$	Shaft Angle Bit 9
14	$\overline{\text{DELA 15}}$	" " " 15

EDGE CONNECTOR E1 - Shaft Speed Output Port

Pin	Name	Signal Function
1		Not Used
2		" "
3		" "
4	SPA Ø	Speed Count Bit Ø
5	SPA 1	" " " 1
6	SPA 2	" " " 2
7	SPA 3	" " " 3
8	SPA 4	" " " 4
9	SPA 5	" " " 5
1Ø	SPA 6	" " " 6
11	SPA 7	" " " 7
12	SPA 8	" " " 8
13	SPA 9	" " " 9
14	SPA 1Ø	" " " 1Ø
15	SPA 11	" " " 11
16	SPA 12	" " " 12
17	SPA 13	" " " 13
18	SPA 14	" " " 14
19	SPA 15	" " " 15
2Ø	SPA 16	" " " 16

EDGE CONNECTOR P1 - Peripheral Bus Interconnector

Pin	Name	Signal Function
1	GND	Supply Common
2		Not Used
3	ZC	Zero Crossing Signal
4	SA1	Shaft Pulse No 1
5	SA2	" " No 2
6	$Ø_1$	2.Ø MHz Processor Clock
7	DELTA1 INT REQ	Transducer 1 Interrupt Request ('RST 3Q')
8		Not Used
9		" "
1Ø		" "
11		" "
12		" "
13		" "
14		" "
15		" "
16		" "
17	$\overline{\text{RESET B}}$	Buffered Reset Signal
18		Not Used
19	-15VDC	-15 volt Source Power
2Ø	-12VDC	-12 volt Source Power

21	+12VDC	+12 volt Source Power
22	+15VDC	+15 volt Source Power
23		Not Used
24		" "
25		" "
26		" "
27		" "
28		" "
29		" "
3Ø		" "
31		" "
32		" "
33		" "
34	DELTA1 STRT	Transducer No 1 Start
35		Not Used
36	DELTA INT ENBL	Transducer Interrupt Enable
37	DELTA AUTO	Auto Mode Operation
38	DELTA1 DONE	Transducer No 1 done (ready)
39		Not Used
4Ø	+5VDC	+5 volt Source Power

8.4.1.4 Shaft Angle and Speed Counter Card No 2

SOCKET J1 - Load Angle Output Port

Pin	Name	Signal Function
1	DELB 8	Shaft Angle Bit 8
2	DELB DATA TMTD	" " Data Transmitted
3	DELB 14	" " Bit 14
4	DELB 13	" " " 13
5	DELB 12	" " " 12
6	DELB 6	" " " 6
7	DELB 3	" " " 3
8	DELB 4	" " " 4
9	DELB 5	" " " 5
1Ø	DELB 7	" " " 7
11	DELB 11	" " " 11
12	DELB 1Ø	" " " 1Ø
13	DELB 9	" " " 9
14	DELB 15	" " " 15

EDGE CONNECTOR E1 - Shaft Speed Output Port

Pin	Name	Signal Function
1		Not Used
2		" "
3		" "
4	SPB Ø	Speed Count Bit Ø
5	SPB 1	" " " 1
6	SPB 2	" " " 2

7	SPB 3	Speed Count Bit 3
8	SPB 4	" " " 4
9	SPB 5	" " " 5
1Ø	SPB 6	" " " 6
11	SPB 7	" " " 7
12	SPB 8	" " " 8
13	SPB 9	" " " 9
14	SPB 1Ø	" " " 1Ø
15	SPB 11	" " " 11
16	SPB 12	" " " 12
17	SPB 13	" " " 13
18	SPB 14	" " " 14
19	SPB 15	" " " 15
2Ø	SPB 16	" " " 16

EDGE CONNECTOR E1 - Peripheral Bus Interconnector

Pin	Name	Signal Function
1	GND	Supply Common
2		Not Used
3	ZC	Zero Crossing Signal
4	SA2	Shaft Pulse No 2
5	SA1	Shaft Pulse No 1
6	$Ø_1$	2.Ø MHz Processor Clock
7	DELTA2 INT REQ	Transducer 2 Interrupt Request ('RST 4Q')
8		Not Used
9		" "
1Ø		" "
11		" "
12		" "
13		" "
14		" "
15		" "
16		" "
17	$\overline{\text{RESET B}}$	Buffered Reset Signal
18		Not Used
19	-15VDC	-15 volt Source Power
2Ø	-12VDC	-12 volt " "
21	+12VDC	+12 volt " "
22	+15VDC	+15 volt " "
23		Not Used
24		" "
25		" "
26		" "
27		" "
28		" "
29		" "
3Ø		" "
31		" "
32		" "
33		" "
34	$\overline{\text{DELTA2 STRT}}$	Transducer No 2 Start
35		Not Used
36	$\overline{\text{DELTA INT ENBL}}$	Transducer Interrupt Enable
37	$\overline{\text{DELTA AUTO}}$	Auto Mode Operation

38	$\overline{\text{DELTA2 DONE}}$	Transducer No 2 done (ready)
39		Not Used
4Ø	+5VDC	+5 volt Source Power

8.4.2 Data Processing

The information from the transducer is available to the micro-processor in binary form. The connections are arranged such that this information will be presented to the micro-processor as 16-bit two's complement (signed) integers. The low-order bits from the shaft angle latch section are not required for accuracy purposes butthe significance of the remaining bits is kept the same causing the number to be quantised in larger steps.

The 16-bit integer representative of shaft angle (N_D) has the following characteristics:

Max value of N_D:	32760 (abs.Lim)	15000 (typ.op.)
Min value of N_D:	Ø (" ")	5000 (" ")
Conversion to radians:	$\alpha = 3.1415 \times 10^{-4} \times N_D$	
Quantisation:	of $N_D = 8$	of $\alpha = 2.5 \times 10^{-3}$ rad

The 16-bit integer representative of the shaft speed (N_S) actually indicates the period of one cycle and has the following characteristics:

Max value of N_S:	32767 (abs.Lim)	20400 (typ.op)
Min value of N_S:	Ø (" ")	19600 (" ")
Conversion to seconds:	$T = 10^{-6} \times N_S$	
Quantisation:	of $N_S = 1$	of $T = 10^{-6}$ secs

8.4.3 Installation Data

Connectors to Peripheral Unit Bus:

Single 4Ø-pin PC Edge Connector. Ø.1 centres
(Compatible with Vero Type 13623-1)

Power Supply Requirements:

+5vdc	±5%	@	35Ø mA typ (6ØØ mA max)
-9vdc	±5%	@	Ø.Ø2 mA typ
-15vdc	±5%	@	3Ø mA typ
+15vdc	±5%	@	2Ø mA typ

Operating Temperature Range:

+Ø°C to +7Ø°C

8.4.4 Component Specification

Integrated Circuits

A2	=	SN74Ø2N	C2	=	SN7475N
A3	=	SN7474N	C3	=	SN7475N
A4	=	SN7474N	C4	=	SN7475N
B1	=	SN74121N	D1	=	SN7493N
B2	=	SN74ØØN	D2	=	SN7493N
B3	=	SN74Ø4N	D3	=	SN7393N
B4	=	SN74ØØN	D4	=	SN7493N

Resistors

R1, R2	1KΩ, $\frac{1}{8}$ Watt, metal oxide
R4 to R6	1KΩ, $\frac{1}{8}$ Watt, metal oxide

Capacitors

C1	22Ø µf, 25 V electrolytic
C2	33ØØ pf
C3	.Ø1 µf, ceramic
C4	75Ø pf
C5,C6	.Ø1 µf, ceramic
C7	22ØØ pf
C8 to C11	.Ø1 µf, ceramic

Sockets

J1	A23 - 2Ø48LC

SECTION 9

TERMINAL VOLTAGE AND E'_q TRANSDUCER UNIT

The Terminal Voltage and E'_q (voltage behind transient reactance) Unit provides the microprocessor with digital readings of these parameters centred on the use of two analogue to digital converter units. It also provides, for development purposes only, an indication to the microprocessor when the terminal voltage falls below a certain pre-set limit.

9.1 GENERAL FUNCTIONAL DESCRIPTION

To convert an analogue value into digital form, it is often necessary to use a voltage analogue to digital converter module as the heart of the system. In such cases, a prior requirement is the conversion of the required parameter into a d.c. voltage of magnitude proportional to that parameter. For the terminal voltage measurement, this primary conversion is performed by a simple three-phase bridge rectifier and filter unit. The filter is a second-order type having a double breakpoint at 8Ø Hz. to give sufficient attenuation of the 3ØØ Hz ripple whilst allowing an adequate response at lower frequencies. The conversion of voltage-behind-transient reactance to a d.c. form is performed by implementation of the relationship:

$$e'_q = e_q + X'_d i_d \qquad \text{(resistance neglected)}$$

requiring the availability of the quadrature axis voltage, e_q, and the direct axis current, i_d. These signals are obtained by a Park's component resolution circuit, the e_q signal being generated in a ring modulator multiplier and the i_d signal being generated in a Hall effect multiplier. The operation of these circuits is well described elsewhere[49] and only the analogue to digital converter card will be considered here.

The functional operation of the unit is relatively simple as

shown in the diagram, Fig 9.1. The two input signals, in d.c. form as previously described, are present at the inputs of the two analogue to digital converter units each of which converts these to digital form under the supervising control of the microprocessor via the control logic. The outputs of the converters are padded out to full 16-bit words and presented to the microprocessor input ports.

9.2 CONTROL AND STATUS REGISTERS

In common with all other transducer units, the Terminal Voltage and E'_q Transducer Unit can be run in several program-selectable modes. These modes are selected by the control bits set in Peripheral Control Register No 1 (Output Port No 1 - see Section 6.3.1)

Bit	Name	Function
Ø	VT STRT	START terminal voltage A/D conversion. The VT DONE flag will be low for 21 µS while conversion is being performed.
1	EQ STRT	START e'_q voltage A/D conversion. The EQ DONE flag will be low for 21 µS while conversion is being performed.
2	VT INT ENBL	Enable the terminal voltage transducer to request an interrupt on completion of the conversion. If accepted, the interrupt is to vector $5Ø_8$ ('RST 5Q').
3	EQ INT ENBL	Enable the e'_q voltage transducer to request an interrupt on completion of the conversion. If accepted, the interrupt is to vector $6Ø_8$ ('RST 6Q').

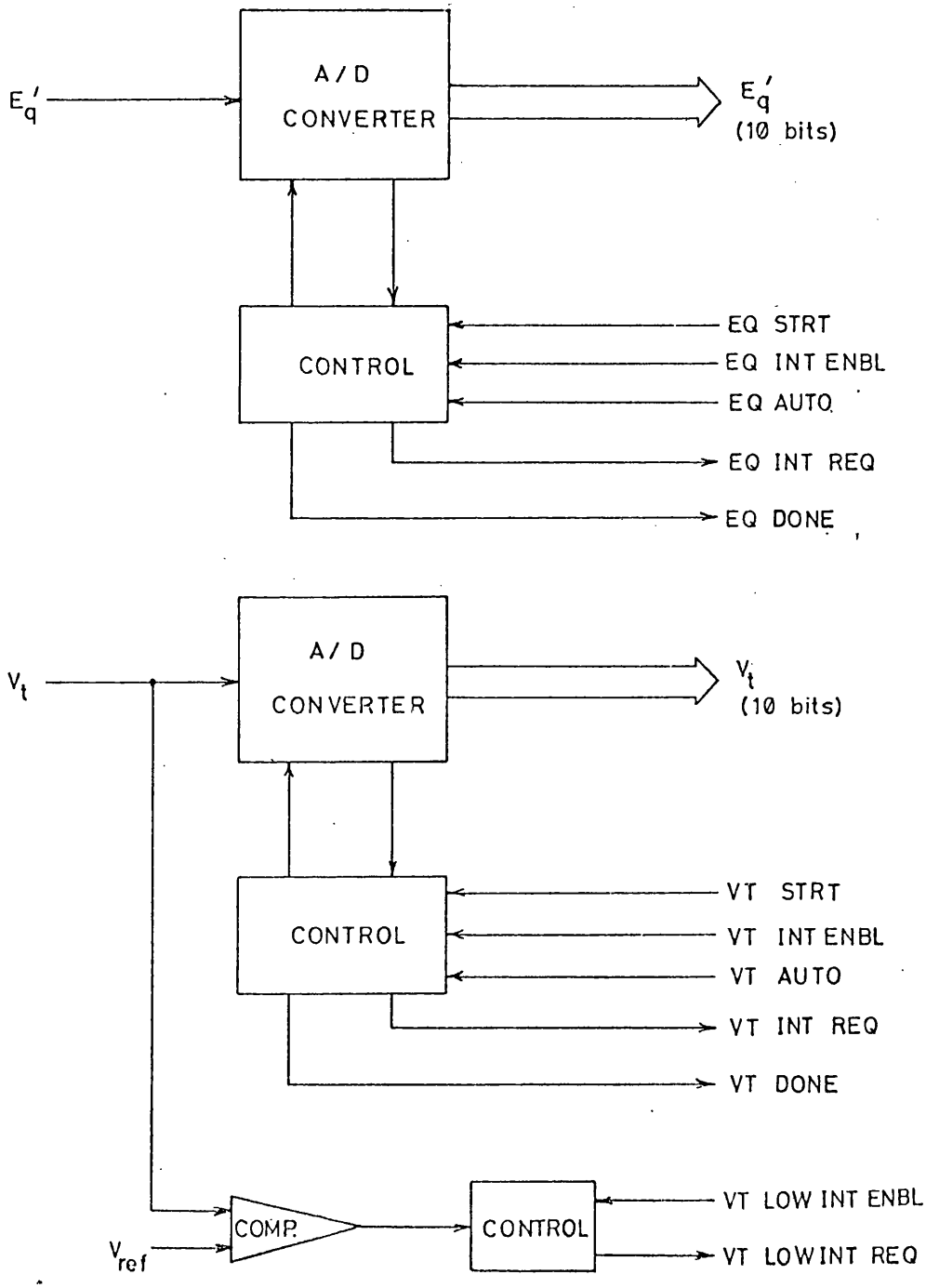


Fig. 9.1 A/D Converter Card Block Schematic

4	VT AUTO	Set the terminal voltage transducer to AUTO mode. Conversion is automatically restarted when data has been transferred to the microprocessor. The ports must be read in the sequence 4, 5 as the conversion is triggered by reading port 5.
5	EQ AUTO	Set the e'_q voltage transducer to AUTO mode. Conversion is automatically restarted when data has been transferred to the microprocessor. The ports must be read in the sequence 6, 7 as the conversion is triggered by reading port 7.

The portion of the circuit which detects a low terminal voltage is controlled by bits in Peripheral Control Register No 2 (Output Port No 2):

Bit	Name	Function
7	VT LOW INT ENBL	Enable the occurrence of a fall in terminal voltage below a preset limit to cause an interrupt request. If accepted, the interrupt is to vector $2Ø_8$ ('RST 2Q').

The status of the transducer unit is displayed in Peripheral Status Register No 1 (Input Port No 1) as follows:

Bit	Name	Function
Ø	VT DONE	Terminal voltage conversion DONE and available for input to the microprocessor. Read only. Causes an

		interrupt request if VT INT ENBL is set.
1	EQ DONE	E'_q voltage conversion DONE and available for input to the microprocessor. Read only. Causes an interrupt request if EQ INT ENBL is set.

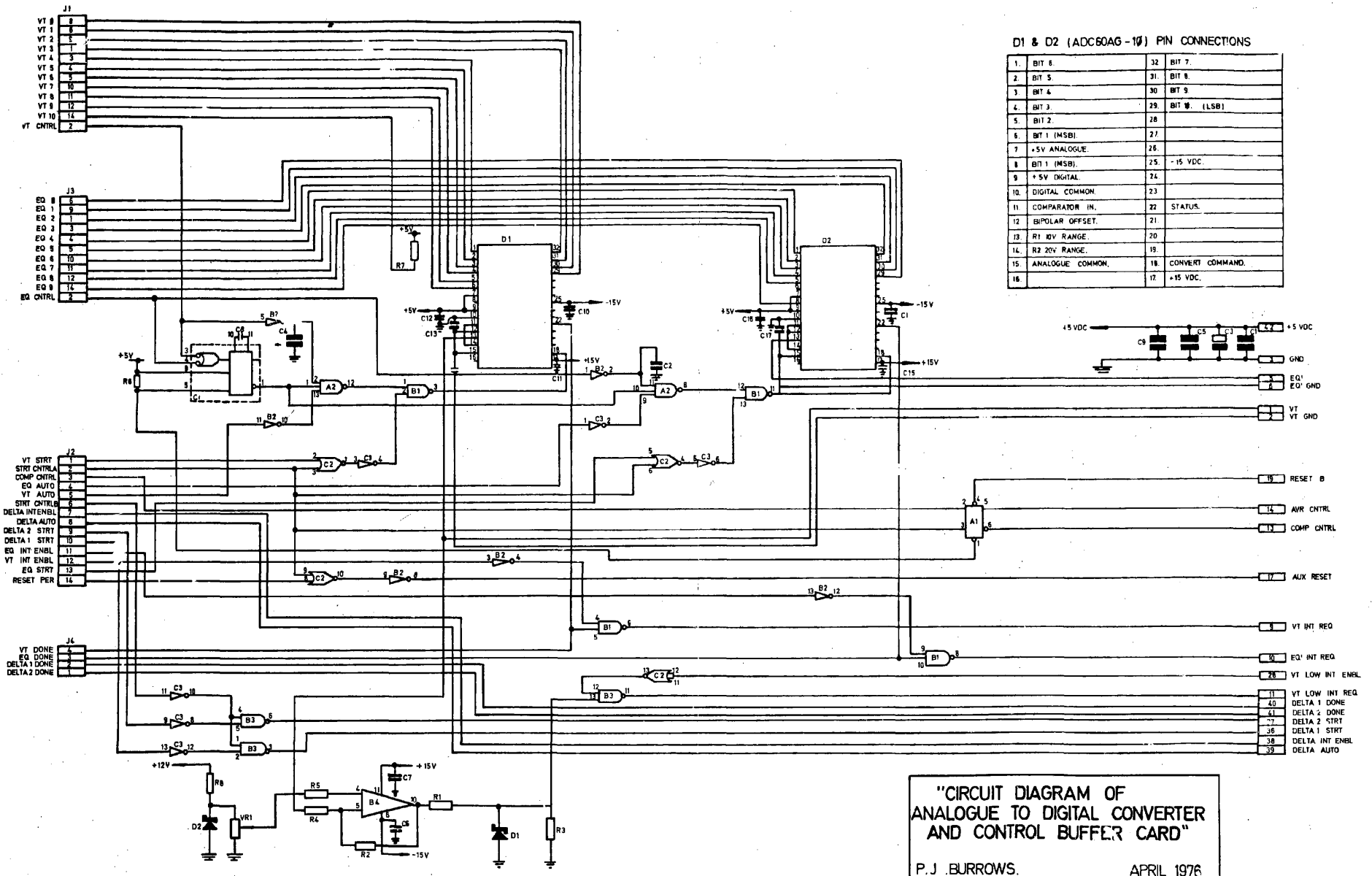
9.3 CIRCUIT DESCRIPTION

The circuit diagram of the A/D converter card is given in Fig 9.2. The operation of both terminal voltage and e'_q converters is similar so only one, the terminal voltage converter, will be described.

Conversion is initiated by one of two means. If a logical '1' is output at bit Ø of Output Port 1, then this appears at pin 1 of J2 (Fig 9.2) as a 'low' signal. This is clocked in the 1,2,3 section of C2 with the DATA TMTD pulse on J2, pin 2 to produce a pulse output necessary to initiate the conversion. This eventually appears at pin 2 of B1 which functions in a negative logic 'OR' mode, such that the pulse will appear on pin 18 of the A/D converter (D1) when pin 1 of B1 is in its normal 'high' state. Alternatively, the start pulse may appear at pin 1 of B1 as a result of the data on socket J1 being read by the microprocessor. This will only occur if pin 13 of A2 is 'high' caused by the presence of a 'low' level at the $\overline{\text{VT AUTO}}$ control, pin 5 of J2. It is necessary that the duration of the DATA TMTD pulse at J1, pin 2 is timed by the monostable C1 to avoid spurious operation as described in Section 6.2.2.

Once conversion has been initiated, pin 22 of the converter will go 'high' informing the microprocessor, via the $\overline{\text{VT DONE}}$ signal on pin 4 of J4, that the device is busy. At the completion of conversion, 21 μS later, this signal will change state to notify the microprocessor. Also, if pin 12 of J2 is low ($\overline{\text{VT INT ENBL}}$) then the 4,5,6 section of B6 will be open and an interrupt request will be sent to the Peripheral Interrupt Priority Card

Fig. 9.2 Circuit Diagram of A/D Converter Card (V_t and E'_q)



D1 & D2 (ADC80AG-10) PIN CONNECTIONS

1.	BIT 8.	32.	BIT 7.
2.	BIT 5.	31.	BIT 6.
3.	BIT 4.	30.	BIT 9.
4.	BIT 3.	29.	BIT 10. (LSB)
5.	BIT 2.	28.	
6.	BIT 1 (MSB).	27.	
7.	+5V ANALOGUE.	26.	
8.	BIT 1 (MSB).	25.	-15 VDC.
9.	+5V DIGITAL.	24.	
10.	DIGITAL COMMON.	23.	
11.	COMPARATOR IN.	22.	STATUS.
12.	BIPOLAR OFFSET.	21.	
13.	R1 10V RANGE.	20.	
14.	R2 20V RANGE.	19.	
15.	ANALOGUE COMMON.	18.	CONVERT COMMAND.
16.		17.	+15 VDC.

"CIRCUIT DIAGRAM OF ANALOGUE TO DIGITAL CONVERTER AND CONTROL BUFFER CARD"

P.J .BURROWS. APRIL 1976

via pin 11 of P1. This may cause the microprocessor to enter an interrupt cycle which will normally cause it to read the converted data available at socket J1. A fresh cycle will be initiated by either of the two methods indicated above.

Additional circuitry on the card is arranged to detect a reduction in terminal voltage. The terminal voltage signal is presented to the non-inverting input of the operational amplifier B4 which is wired as a comparator. The reference voltage is derived from the preset potentiometer VR1 and the output, pin 1Ø, will normally be high when the terminal voltage signal is higher than this preset level. If the terminal voltage falls below this level then the comparator output will change state applying a logical 'Ø' to pin 13 of B3. If the gate is open due to the presence of a 'low' $\overline{\text{VT LOW INT ENBL}}$ signal on pin 28 of P1 then a VT LOW INT REQ signal will be passed to the Peripheral Interrupt Priority Card.

9.4 UTILIZATION

9.4.1 Input and Output Connections

All inputs and outputs of the Terminal Voltage and E'_q Transducer Unit are TTL compatible with the exception of the analogue inputs on pins 1 and 5 of the connector P1.

SOCKET J1 - Terminal Voltage Digital Output

Pin	Name	Function
1	$\overline{\text{VT3}}$	Terminal Voltage Bit 3 * (+Ø.Ø39Ø6v)
2	$\overline{\text{VT DATA TMTD}}$	" " Data Transmitted
3	$\overline{\text{VT4}}$	" " Bit 4 * (+Ø.Ø7813v)
4	$\overline{\text{VT5}}$	" " Bit 5 * (+Ø.1562v)
5	$\overline{\text{VT6}}$	" " Bit 6 * (+Ø3.25v)
6	$\overline{\text{VT1}}$	" " Bit 1 * (+Ø.ØØ977v)
7		Not Used
8	$\overline{\text{VTØ}}$	Terminal Voltage Bit Ø * (+Ø.ØØ488v)
9	$\overline{\text{VT2}}$	" " Bit 2 * (+Ø.Ø1953v)
1Ø	$\overline{\text{VT7}}$	" " Bit 7 * (+Ø.625v)
11	$\overline{\text{VT8}}$	" " Bit 8 * (+1.25v)
12	$\overline{\text{VT9}}$	" " Bit 9 * (+2.5v)
13		Not Used
14	$\overline{\text{VT1Ø}}$	Terminal Voltage Bit 1Ø * (-5.Øv)

*Note: A/D Converter data sheet PDS-333 uses the convention MSB - Bit 1 and LSB - Bit 1Ø. The convention used here confirms with I8Ø8Ø and PDP-11/2Ø standard practice, i.e. LSB - Bit Ø and MSB - Bit 1Ø.

SOCKET J2 - Peripheral Control Input

Pin	Name	Function
1	$\overline{\text{VT STRT}}$	Start Terminal Voltage A/D Conversion
2	$\overline{\text{CNTRL1 DATA RDY}}$	Control Register No 1 Data Ready
3	$\overline{\text{COMP CNTRL}}$	Computer controlling field voltage
4	$\overline{\text{EQ AUTO}}$	E'_q A/D conversion Auto mode
5	$\overline{\text{VT AUTO}}$	Terminal voltage A/D conversion Auto mode
6	$\overline{\text{CNTRL2 DATA RDY}}$	Control Register No 2 Data Ready
7	$\overline{\text{DELTA INT ENBL}}$	Load Angle transducer interrupt enable
8	$\overline{\text{DELTA AUTO}}$	Load Angle transducer Auto mode
9	$\overline{\text{DELTA2 STRT}}$	Start Load Angle transducer No 2
1Ø	$\overline{\text{DELTA1 STRT}}$	" " " " No 1
11	$\overline{\text{EQ INT ENBL}}$	E'_q A/D converter interrupt enable
12	$\overline{\text{VT INT ENBL}}$	Terminal Voltage A/D converter interrupt
13	$\overline{\text{EQ STRT}}$	Start E'_q A/D conversion /enable
14	PER RESET	Reset microprocessor peripherals

SOCKET J3 - Eq' Voltage Digital Output

Pin	Name	Function
1	$\overline{\text{EQ2}}$	Eq' Voltage Bit 2 * (+Ø.Ø1953v)
2	$\overline{\text{EQ DATA TMTD}}$	Eq' data transmitted
3	$\overline{\text{EQ3}}$	Eq' Voltage Bit 3 * (+Ø.Ø39Ø6v)
4	$\overline{\text{EQ4}}$	Eq' " Bit 4 * (+Ø.Ø7813v)
5	$\overline{\text{EQ5}}$	Eq' " Bit 5 * (+Ø.15626v)
6	$\overline{\text{EQØ}}$	Eq' " Bit Ø * (+Ø.ØØ488v)
7		Not Used
8		" "
9	$\overline{\text{EQ1}}$	Eq' Voltage Bit 1 * (+Ø.ØØ977v)
1Ø	$\overline{\text{EQ6}}$	Eq' " Bit 6 * (+Ø.3125v)
11	$\overline{\text{EQ7}}$	Eq' " Bit 7 * (+Ø.625v)
12	$\overline{\text{EQ8}}$	Eq' " Bit 8 * (+1.25v)
13		Not Used
14	$\overline{\text{EQ9}}$	Eq' Voltage Bit 9 * (-2.5v)

*See note relating to Socket J1.

SOCKET J4 - Peripheral Status Output

Pin	Name	Function
1	DELTA2 DONE	Load Angle transducer No 2 done (ready)
2	DELTA1 DONE	" " " No 1 " "
3	EQ DONE	Eq' voltage A/D conversion done (ready)
4	VT DONE	Terminal voltage A/D conversion done (ready)

EDGE CONNECTOR P1 - Peripheral Bus Interconnector

Pin	Name	Function
1	VT	Terminal Voltage Analogue Input
2	VT GND	Terminal Voltage Signal Ground
3	GND	Supply Common
4		Not Used
5	EQ	Eq' Voltage analogue input
6	EQ GND	Eq' Voltage Signal Ground
7	XX	Reference Slot
8		Not Used
9	VT INT REQ	Terminal Voltage A/D Interrupt Request ('RST 5Q')
10	EQ INT REQ	Eq' Voltage A/D Interrupt Request ('RST 6Q')
11	VT LOW INT REQ	Low Terminal Voltage Interrupt Request ('RST 2Q')
12		Not Used
13	COMP GATE	Computer in control of Field Voltage
14	AVR GATE	
15		Not Used
16		" "
17	AUX RESET	Auxilary Reset Output
18		Not Used
19	RESET B	Buffered Peripheral Reset
20		Not Used
21	-15 VDC	-15 volt Source Power
22	-12 VDC	-12 " " "
23	+12 VDC	+12 " " "
24	+15 VDC	+15 " " "
25		Not Used
26		" "
27		" "
28	VT LOW INT ENBL	Low Terminal Voltage Interrupt Enable
29		Not Used
30		" "
31		" "
32		" "
33		" "
34		" "
35		" "
36	DELTA1 STRT	Start Load Angle transducer No 1
37	DELTA2 STRT	Start Load Angle transducer No 2
38	DELTA INT ENBL	Load Angle transducer Interrupt Enable
39	DELTA AUTO	Load Angle transducer Auto mode

4Ø	$\overline{\text{DELTA1 DONE}}$	Load Angle transducer No 1 done (ready)
41	$\overline{\text{DELTA2 DONE}}$	" " " No 2 " "
42	+5 VDC	+5 volt Source Power
43		Not Used

9.4.2 Data Processing

The connections to the microprocessor are arranged such that the information from the transducers is read as two 16-bit binary numbers in two's-complement form. The 16-bit integer representative of terminal voltage (N_T) has the following characteristics:

Maximum value of N_T : 32736 (abs lim) 2836Ø (typ op)
Minimum value of N_T : Ø (" ") 64ØØ (" ")
Conversion to volts : $V_t = N_T/128.Ø$
Quantisation: of N_T = 32 of V_T = Ø.25 volts

The 16-bit integer representative of voltage behind transient reactance (N_E) has the following characteristics:

Maximum value of N_E : 327Ø4 (abs lim) 131Ø7 (typ op)
Minimum value of N_E : -32768 (" ") Ø (typ op)
Conversion to per unit form : $e'_q = N_E/131Ø7.2$
Quantisation: of N_E = 64 of $e'_q = 4.88 \times 1Ø^{-3}$

9.4.3 Installation Data

Connection to Peripheral Unit Bus:-

Single 43-pin PC Edge Connector. Ø.1 centres
(Compatible with Vero type 14-Ø2Ø7E)

Power Supply Requirements:-

+5 vdc ±5% @ 25Ø mA typ (35Ø mA max)
-15 vdc ±5% @ 7Ø mA typ
+15 vdc ±5% @ 6Ø mA typ

Operating Temperature Range:-

$+Ø^{o}C$ to $+7Ø^{o}C$

9.4.4. Component Specification

Integrated Circuits

A1	=	SN7474N	C1	=	SN7412N
A2	=	SN741ØN	C2	=	SN74Ø2N
B1	=	SN74ØØN	C3	=	SN74Ø4N
B2	=	SN74Ø4N	D1	=	ADc8ØAG-1Ø
B3	=	SN74ØØN	D2	=	ADC8ØAG-1Ø
B4	=	SN727Ø9N			

Resistors

R1	47Ø Ω, $\frac{1}{8}$ W, metal oxide
R2	1MΩ, " "
R3	4.7 KΩ " "
R4,R5	1Ø KΩ " "
R6	1 KΩ " "
R7	1 KΩ " "
VR1	1Ø KΩ, skeleton preset

Capacitors

C1	.Ø1 µf ceramic	C10,C11	1 µf
C2	22ØØ pf	C12	.Ø1 µf ceramic
C3	8Ø µf, 25 V, electrolytic	C13	Ø.1 µf ceramic
C4	22ØØ pf	C14,C15	1 µf
C5 to C7	Ø.1 µf ceramic	C16	Ø.Ø1 µf ceramic
C8	68Ø pf	C17	Ø.1 µf ceramic
C9	.Ø1 µf ceramic		

Sockets

J1	A23-2Ø48LC
J2	A23-2Ø48LC
J3	A23-2Ø48LC
J4	A23-2Ø7ØPS

Diodes

D1	4.7, $\frac{1}{4}$ W, Zener	D2	5.6 V, $\frac{1}{4}$ W, Zener

SECTION 1Ø

FIELD VOLTAGE A/D TRANSDUCER UNIT

The Field Voltage A/D Transducer Unit provides the microprocessor with digital readings of the synchronous machine field voltage regardless of whether it is generated by analogue or digital means. This facility is particularly useful for comparative tests of analogue and digital control systems.

1Ø.1 GENERAL FUNCTIONAL DESCRIPTION

The circuit function is centred around an analogue to digital converter module and is almost identical in operation to the terminal voltage transducer which has been previously described. (Section 9.1). The unit also contains an unused analogue to digital converter sub-section available for future expansion.

1Ø.2 CONTROL AND STATUS REGISTERS

The unit may be used in several program selectable modes in a similar manner to other transducer units. These modes are selected by the control bits set in Peripheral Control Register No 2 (Output Port No 2):-

Bit	Name	Function
4	VFI STRT	START field voltage A/D conversion. The VFI DONE flag will be low for 21 µS while conversion is being performed.
5	VFI INT ENBL	Enable the field voltage transducer to request an interrupt on completion of the conversion. If accepted the interrupt is to vector $7Ø_8$ ('RST 7Q').

6	VFI AUTO	Set the field voltage transducer to AUTO mode. Conversion is automatically restarted when data has been transferred to the microprocessor. The ports must be read in the sequence 2,3 as the conversion is triggered by reading port 3.

The status of the transducer unit is displayed in the Peripheral Status Register No 1 (Input Port No 1):-

Bit	Name	Function
4	VFI DONE	Field Voltage Conversion DONE and available for input to the microprocessor. Read only. Causes an interrupt request if VT INT ENBL is set.

1Ø.3 CIRCUIT DESCRIPTION

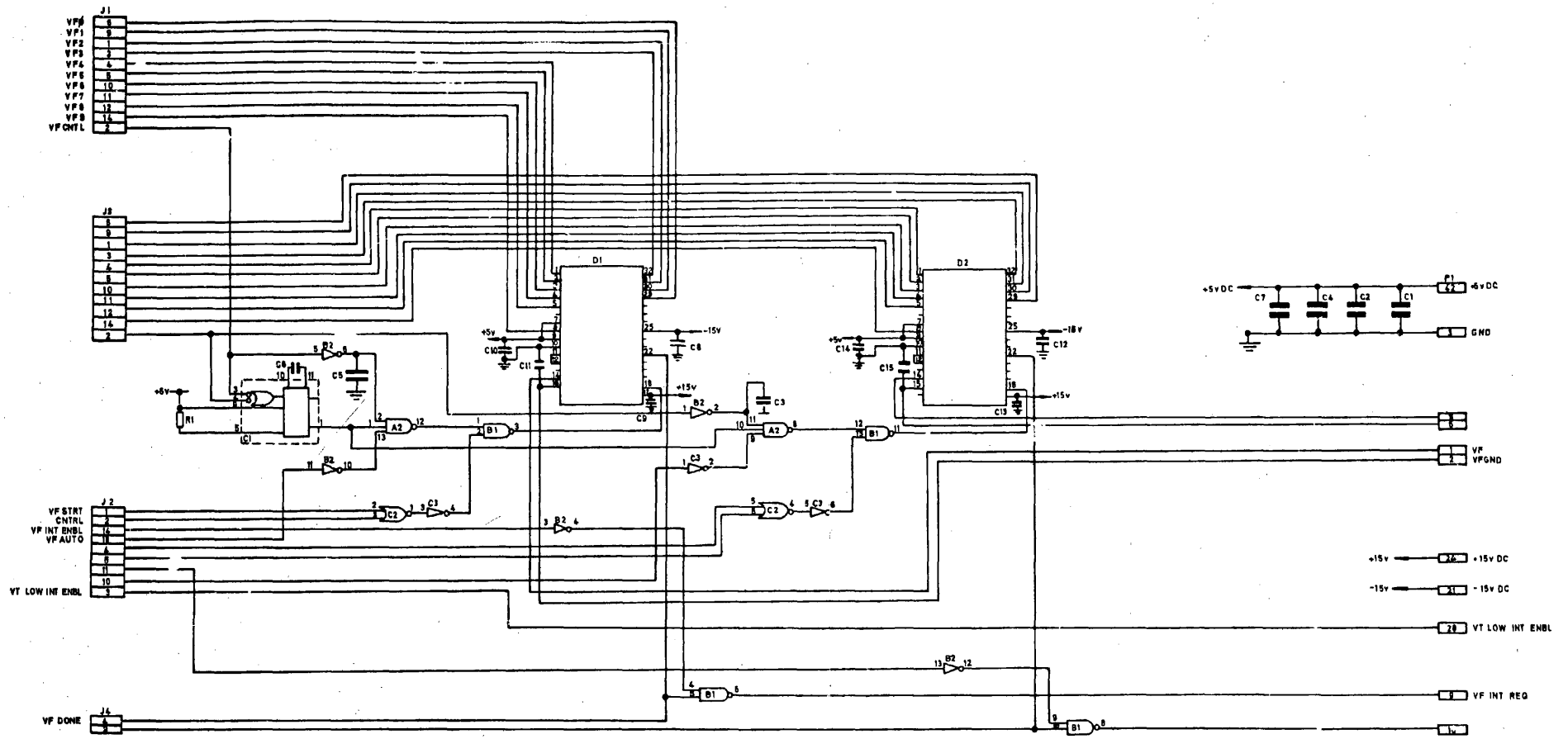
The circuit diagram of the A/D converter card is given in Fig 1Ø.1. As shown, the card contains the facility for two A/D converter modules, one of which is available for future system expansion. Each module is identical in operation with those in the Terminal Voltage and E'_q Transducer Unit and reference should be made to Section 9.3 for a description of the circuit operation.

1Ø.4 UTILIZATION

1Ø.4.1 Input and Output Connections

All inputs and outputs of the Field Voltage A/D Converter Card are TTL compatible with the exception of the analogue inputs on pins 1 and 5 of connector P1.

Fig. 10.1 Circuit Diagram of A/D Converter Card (V_f)



ANALOGUE / DIGITAL CONVERTER
(CIRCUIT DIAGRAM) CARD.

P.J. BURROWS AUGUST 1976

SOCKET J1 - Field Excitation Voltage Digital Output

Pin	Name	Signal Function
1	$\overline{\text{VFI2}}$	Field Voltage Bit 2 (+Ø.Ø7813v)
2	VF DATA TMTD	" " Data Transmitted
3	$\overline{\text{VFI3}}$	" " Bit 3 (+Ø.15625v)
4	$\overline{\text{VFI4}}$	" " " 4 (+Ø.3125v)
5	$\overline{\text{VFI5}}$	" " " 5 (+Ø.625v)
6	$\overline{\text{VFIØ}}$	" " " Ø((+Ø.Ø1953v)
7		Not Used
8		" "
9	$\overline{\text{VFI1}}$	Field Voltage Bit 1 (+Ø.Ø39Øv)
1Ø	$\overline{\text{VFI6}}$	" " " 6 (+1.25v)
11	$\overline{\text{VFI7}}$	" " " 7 (+2.55v)
12	$\overline{\text{VFI8}}$	" " " 8 (+5.Øv)
13		Not Used
14	$\overline{\text{VFI9}}$	Field Voltage Bit 9 (-1Ø.Øv)

SOCKET J2 - Peripheral Control Input

Pin	Name	Signal Function
1	$\overline{\text{VFI STRT}}$	Start Field Voltage A/D Conversion
2	$\overline{\text{CNTRL2 DATA RDY}}$	Control Register No 2 Data Ready
3	$\overline{\text{VT LOW INT ENBL}}$	Low Terminal Voltage Interrupt Enable
4	$\overline{\text{AUX STRT}}$	Start Spare Voltage A/D conversion
5	AUX CNTRL DATA RDY	Auxilary Control Register Data Ready
6		
7		
8		
9		
1Ø	$\overline{\text{AUX AUTO}}$	Spare Voltage A/D Auto Conversion Mode
11	AUX INT ENBL	Spare Voltage A/D Interrupt Enable
12		
13	$\overline{\text{VFI AUTO}}$	Field Voltage A/D Auto Conversion Mode
14	VFI INT ENBL	Field Voltage A/D Interrupt Enable

SOCKET J3 - Spare Voltage Digital Output

Pin	Name	Signal Function
1	$\overline{\text{AUX2}}$	Spare Voltage Bit 2 (+Ø.Ø7813v)
2	AUX DATA TMTD	Spare Voltage Data Transmitted
3	$\overline{\text{AUX3}}$	" " Bit 3 (+Ø.15625v)
4	$\overline{\text{AUX4}}$	" " " 4 (+Ø.3125v)
5	$\overline{\text{AUX5}}$	" " " 5 (+Ø625v)
6	$\overline{\text{AUXØ}}$	" " " Ø (+Ø.Ø1953v)
7		Not Used
8		" "
9	$\overline{\text{AUX1}}$	Spare Voltage Bit 1 (+Ø.Ø39Ø6v)

1Ø	$\overline{\text{AUX6}}$	Spare Voltage Bit 6 (+1.25v)
11	$\overline{\text{AUX7}}$	" " " 7 (+2.5v)
12	$\overline{\text{AUX8}}$	" " " 8 (+5.0v)
13		Not Used
14	$\overline{\text{AUX9}}$	Spare Voltage Bit 9 (-1Ø.Øv)

SOCKET J4 - Peripheral Status Output

Pin	Name	Signal Function
1		Not Used
2		" "
3	$\overline{\text{AUX DONE}}$	Spare Voltage A/D Conversion Done(ready)
4	$\overline{\text{VFI DONE}}$	Field Voltage A/D Conversion Done(ready)

EDGE CONNECTOR P1 - Peripheral Bus Interconnector

Pin	Name	Signal Function
1	VF	Field Voltage Analogue Input
2	VF GND	Field Voltage Signal Ground
3	GND	Supply Common
4		Not Used
5	AUX	Spare Voltage Analogue Input
6	AUX GND	Spare Voltage Signal Ground
7	XX	Reference Slot
8		Not Used
9	VF INT REQ	Field Voltage Interrupt Request ('RST 7Q')
1Ø	AUX INT REQ	Spare " " "
11		Not Used
12		" "
13		" "
14		" "
15		" "
16		" "
17		" "
18		" "
19		" "
2Ø		" "
21	-15 VDC	-15 Volt Source Power
22	-12 VDC	-12 " " "
23	+12 VDC	+12 " " "
24	+15 VDC	+15 " " "
25		Not Used
26		" "
27		" "
28	$\overline{\text{VT LOW INT ENBL}}$	Low Terminal Voltage Interrupt Enable
29		Not Used
3Ø		" "
31		" "
32		" "
33		" "
34		" "

35		Not Used
36		" "
37		" "
38		" "
39		" "
4Ø		" "
41		" "
42	+5 VDC	+5 Volt Source Power
43		Not Used

1Ø.4.2 Data Processing

The connections to the microprocessor are arranged such that the information from the transducers is read as two 16-bit binary numbers in two's-complement form. The 16-bit integer representative of field voltage (N_F) has the following characteristics:

Maximum value of N_F: 327Ø4 (abs lim) 223ØØ (typ op)
Minimum value of N_F: -32768 (abs lim) -223ØØ (typ op)
Conversion to per unit form: $V_f = N_F / 3276.8$
Quantisation: of N_F = 64 of V_F = Ø.Ø195 p.u.

1Ø.4.3 Installation Data

Connection to Peripheral Unit Bus:-

Single 43-pin PC Edge Connector. Ø.1 centres
(Compatible with Vero type 14-Ø2Ø7E)

Power Supply Requirements:-

+5 vdc ±5% @ 225 mA typ (3ØØ mA max)
+15 vdc ±5% @ 40 mA typ
-15 vdc ±5% @ 40 mA typ

Operating Temperature Range:-

+Ø°C to +7Ø°C

1Ø.4.4 Component Specifications

Integrated Circuits

A2	=	SN741ØN	C3	=	SN74Ø4N
B1	=	SN74ØØN	D1	=	ADC8ØAG-1Ø
B2	=	SN74Ø4N	D2	=	ADC8ØAG-1Ø
C1	=	SN7412lN			
C2	=	SN74Ø2N			

Resistors

R1 = 1 KΩ $\frac{1}{8}$ Watt metal oxide

Capacitors

C1	8Ø µF, 25 V, electrolytic
C2	Ø.1 µF, ceramic
C3	22ØØ pF
C4	.Ø1 µF, ceramic
C5	22ØØ pF
C6	68Ø pF
C7	.Ø1 µF, ceramic
C8	1 µF
C9	1 µF
C1Ø	.Ø1 µF, ceramic
C11	Ø.1 µF, ceramic
C12	1 µF
C13	1 µF
C14	.Ø1 µF, ceramic
C15	Ø.1 µF, ceramic

Sockets

J1	A23-2Ø48LC
J2	A23-2Ø48LC
J3	A23-2Ø48LC
J4	A23-2Ø7ØPS

SECTION 11

FIELD VOLTAGE D/A TRANSDUCER UNIT

The Field Voltage D/A Transducer Unit provides the facility for the microprocessor to control the micro-machine excitation via the time-constant regulator unit.

11.1 GENERAL FUNCTIONAL DESCRIPTION

The Field Voltage D/A Transducer Unit is based on a standard digital to analogue converter module. As the output from this converter will be added into the input of the high-gain AVR controlling the micro-machine, it is necessary that the output signal is as noise-free as possible. In particular, it is important to remove any ground-loop problems and this is achieved by optical isolation of the digital input and the analogue output. The power supply for the converter module is also taken from the 'analogue'system such that the output voltage is referenced to that part of the overall laboratory system.

11.2 CIRCUIT DESCRIPTION

The circuit diagram of the digital to analogue converter card is given in Fig 11.1. The 12-bit digital output from the microprocessor is applied to the card at the socket J1 and drives the LEDs in the driver portions of the ILQ 74 optical isolator modules. The outputs from the optical isolator are then applied to the digital inputs of the D/A converter, B3, the high order bit being inverted to transform the two's complement code of the microprocessor to the offset binary code of converter.

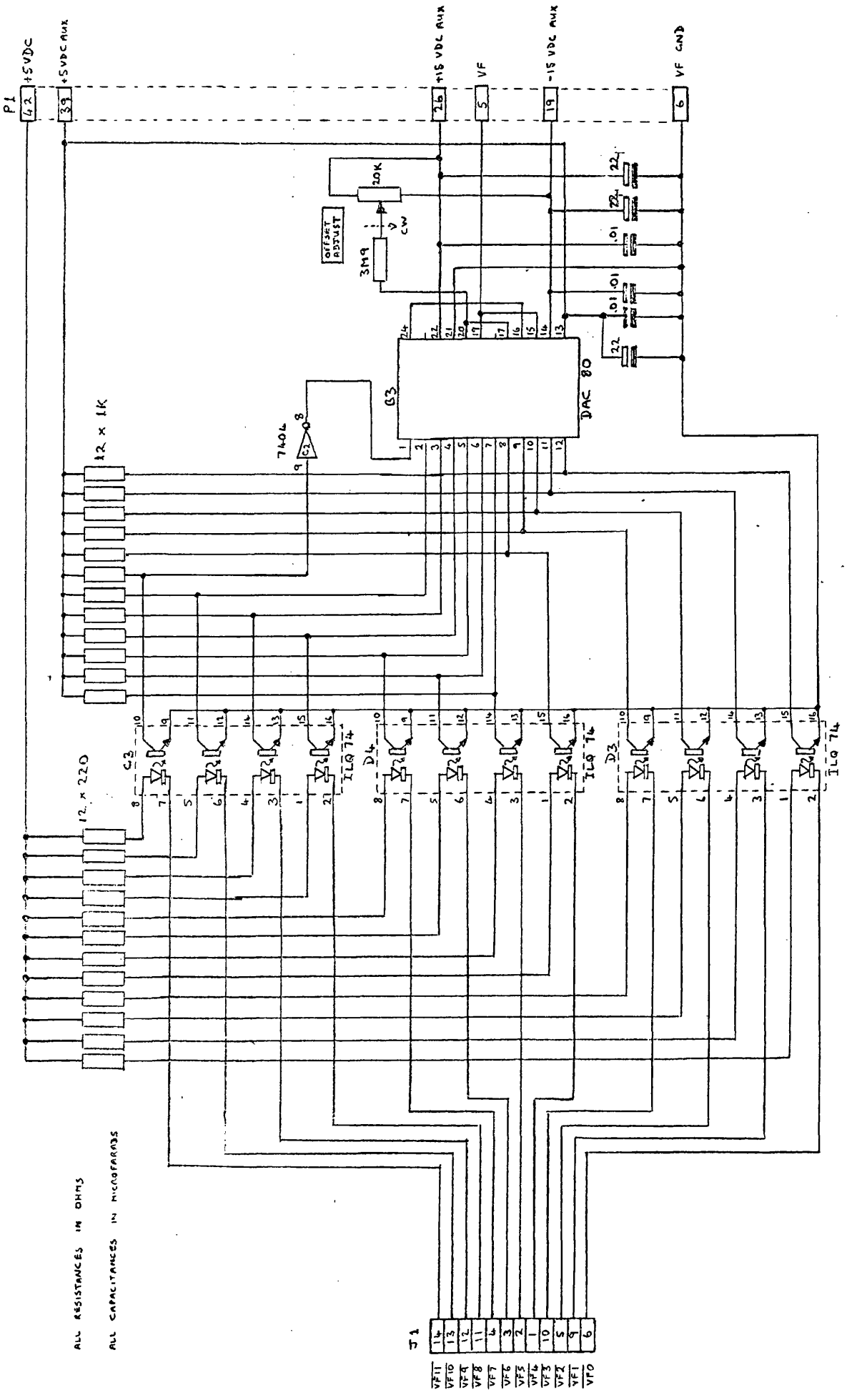


Fig 11.1 Digital/Analogue Transducer Unit Circuit Diagram

11.3 UTILIZATION

11.3.1 Input and Output Connectors

The digital inputs to the card are TTL compatible but represent eight standard TTL loads because of the current requirements of the optical isolators.

SOCKET J1 - Field Excitation Voltage Digital Input

Pin	Name	Signal Function
1	$\overline{VF4}$	Field Voltage Bit 4 * (+Ø.Ø7813v)
2	$\overline{VF5}$	" " Bit 5 * (+Ø.15625v)
3	$\overline{VF6}$	" " Bit 6 * (+Ø.3125v)
4	$\overline{VF7}$	" " Bit 7 * (+Ø.625v)
5	$\overline{VF2}$	" " Bit 2 * (+Ø.Ø1953v)
6	$\overline{VFØ}$	" " Bit Ø * (+Ø.ØØ488v)
7		Not Used
8		" "
9	$\overline{VF1}$	Field Voltage Bit 1 * (+Ø.ØØ977v)
1Ø	$\overline{VF3}$	" " Bit 3 * (+Ø.Ø39Ø6v)
11	$\overline{VF8}$	" " Bit 8 * (+1.25v)
12	$\overline{VF9}$	" " Bit 9 * (+2.5v)
13	$\overline{VF1Ø}$	" " Bit 1Ø * (+5.Øv)
14	$\overline{VF11}$	" " Bit 11 * (-1Ø.Øv)

*Note: D/A Converter data sheet PDS-322B uses the convention MSB - Bit 1 and LSB - Bit 1Ø. The convention used here conforms with I8Ø8Ø and PDP-11/2Ø standard practice, i.e. LSB - Bit Ø and MSB - Bit 9.

EDGE CONNECTOR P1 - Peripheral Bus Interconnector

Pin	Name	Signal Function
1		Not Used
2		" "
3	GND	Supply Common
4		Not Used
5	VF	Field Voltage Analogue Output
6	VF GND	Field Voltage Signal Ground
7	XX	Reference Slot
8		Not Used
9		" "
1Ø		" "

11		Not Used
12		" "
13		" "
14		" "
15		" "
16		" "
17		" "
18		" "
19	-15 VDC AUX	-15 volt Auxilary Source Power
2Ø		Not Used
21	-15 VDC	-15 Volt Source Power
22	-12 VDC	-12 " " "
23	+12 VDC	+12 " " "
24	+15 VDC	+15 " " "
25		Not Used
26	+15 VDC AUX	+15 Volt Auxilary Source Power
27		Not Used
28		" "
29		" "
3Ø		" "
31		" "
32		" "
33		" "
34		" "
35		" "
36		" "
37		" "
38		" "
39	+5 VDC AUX	+5 Volt Auxilary Source Power
4Ø		Not Used
41		" "
42	+5 VDC	+5 Volt Source Power
43		Not Used

11.3.2 Data Processing

The connections between the microprocessor and the counter are arranged such that the information to the converter may be output as a 16-bit binary integer in two's complement form. This integer (N_X) has the following characteristics:-

Maximum value of N_X: 32767 (abs lim) 225ØØ (typ op)
Minimum value of N_X: -32768 (" ") -225ØØ (" ")
Conversion to per unit form: $V_F = N_T / 3276.8$
Quantisation: of N_X = 16 of V_F = $4.88 \times 1Ø^{-3}$

11.3.3 Installation Data

Connection to Peripheral Unit Bus:-

Single 43-pin PC Edge Connector. Ø.1 centres
(Compatible with Vero type 14-Ø2Ø7E)

Power Supply Requirements:-

Digital system:	+5 vdc	±5% @	2ØØ mA typ
Analogue system:	+5 vdc	±5% @	7Ø mA typ
	+15 vdc	±5% @	25 mA typ
	-15 vdc	±5% @	25 mA typ

Operating Temperature Range:-

+Ø°C to +7Ø°C

11.3.4 Component Specifications

Integrated Circuits

B3	=	DAC8Ø-CBI-V	D3	=	ILQ 74
C2	=	SN74Ø4	D4	=	ILQ 74
C3	=	ILQ 74			

Resistors

R1 to R12	22Ø Ω, $\frac{1}{8}$ W, metal oxide
R13 to R24	1 KΩ, $\frac{1}{8}$ W, metal oxide
R25	3.9 MΩ, $\frac{1}{8}$ W, " "
VR1	20 KΩ, multi-term, preset

Capacitors

C1 to C3	22 µF, 25v, electrolytic
C4 to C6	Ø.Ø1 µF ceramic

SECTION 12

POWER SUPPLIES AND SYSTEM MAINFRAME

12.1 GENERAL DESCRIPTION

The microprocessor and peripherals are mounted in a single rack-frame unit adjacent to the micro-machine model. One rack unit houses the interface, microprocessor CPU, memory and input/output cards while the other houses all the transducer cards. The power supplies are mounted in the base of the rack unit and have additional capacity for future extensions.

12.2 POWER SUPPLIES

The power supplies are commercial units with the exception of the -9 volt supply for which no commercial unit was conveniently available. The circuit diagram of this supply is given in Fig 12.1 and the output characteristics given in Fig 12.2. The full power supply facility is as follows:-

+15 Ø -15 VDC	Type:	OA3
	Maximum Current:	25Ø mA
	Current Limit:	35Ø mA
AC Supply Current: Ø.13 A max	Short-Cct Current:	1ØØ mA
+12 Ø -12 VDC	Type:	MTA
	Maximum Current:	5ØØ mA
	Current Limit:	55Ø mA
AC Supply Current: Ø.18 A max	Short-Cct Current:	25Ø mA
-9 VDC	Type:	In-house
	Maximum Current:	2.Ø A
	Current Limit:	2.25 A
AC Supply Current: Ø.23 A max	Short-Cct Current:	1.Ø A

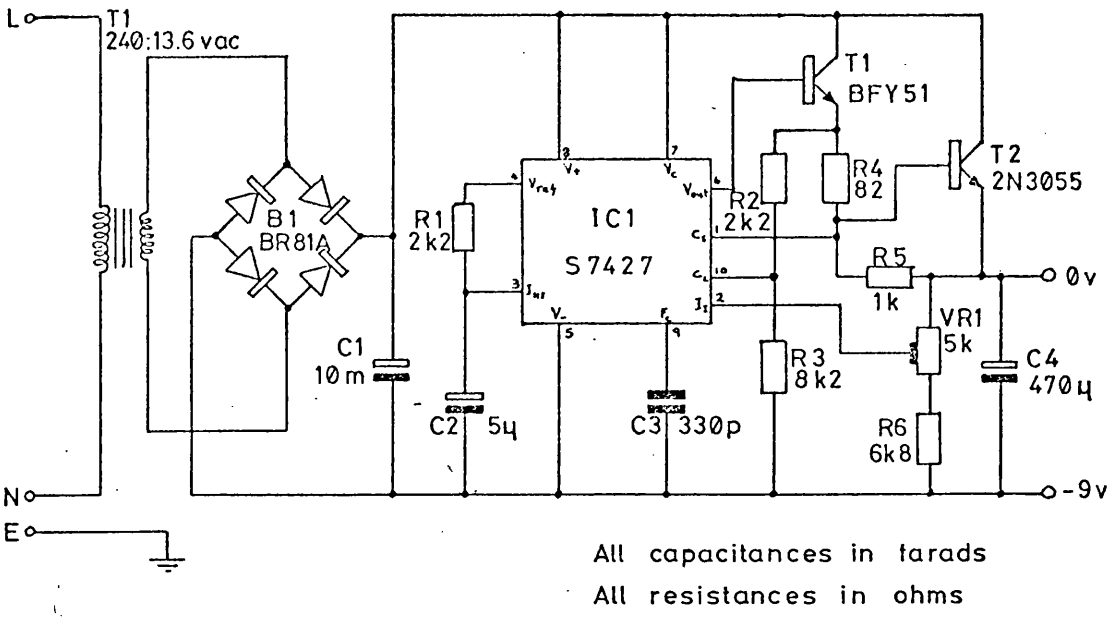


Fig. 12.1 9 Volt Power Supply Circuit Diagram

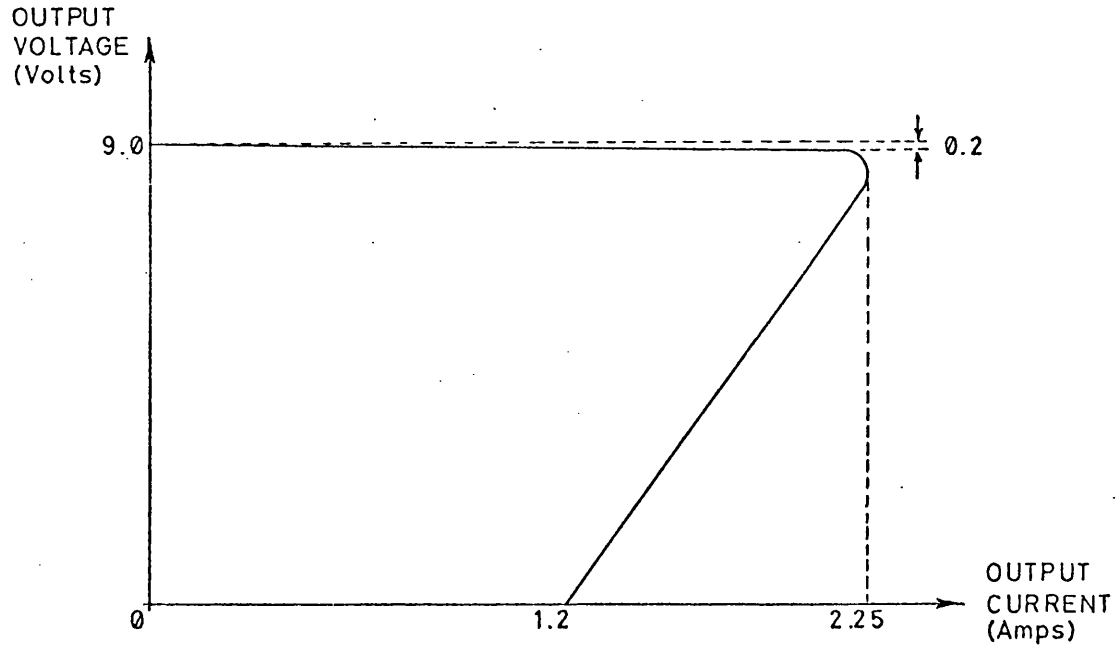


Fig. 12.2 9 Volt Supply Output Characteristic

-5 VDC	Type:	IC4ØØ/6
	Maximum Current:	4.Ø A
	Current Limit:	4.5 A
AC Supply Current: Ø.4Ø A max	Short-Cct Current:	1.5 A
+5 VDC	Type:	'S' Type
	Maximum Current:	2Ø A
	Current Limit:	22 A
AC Supply Current: 1.8Ø A max	Short-Cct Current:	4 A

Total AC Supply Current: 2.75 A Max

12.3 MAIN FRAME

There are two power supply connectors to the Peripheral Unit which are connected as follows:-

PERIPHERAL CARD FRAME MAIN CONNECTOR Plug No P1

Pin	Name	Signal Function
1	-15 VDC	-15 volt Source Power
2	-12 VDC	-12 " " "
3	-9 VDC	-9 volt Source Power
4	-5 VDC	-5 " " "
5	GND	Supply Common
6	+5 VDC	+5 volt Source Power
7	+12 VDC	+12 volt Source Power
8	+15 VDC	+15 " " "
9		Not Used
1Ø	$\overline{\text{RESET}}$	Reset Signal from micrproccssor bus
11	VIN	Infinite Bus Neutral
12	VIR	Infinite Bus 'Red' Phase Signal

PERIPHERAL CARD FRAME AUXILARY CONNECTOR Plug No P2

Pin	Name	Signal Function
1		Not Used
2	-15 VDC AUX	-15 volt Auxilary Source Power
3	+5 VDC AUX	+5 volt Auxilary Source Power
4	+15 VDC AUX	+15 volt Auxilary Source Power

SECTION 13

REAL-TIME SYSTEM AND DEVICE HANDLERS

The following sections (13 to 18) describe the major software components of the digital control system. These include both real-time control programs and subroutines, and program development and support facilities. All of these function under the supervision of the real-time monitor system, which is based on the Digital Equipment RT-11 Monitor[45,48]. This system has been modified, mainly by device handler changes and system program additions, to provide an operating environment which allows fast and flexible development of real-time control programs for the micromachine generating system.

13.1 THE RT-11 SYSTEM

RT-11 is a single-user programming and operating system designed for the PDP-11 series of computers. It provides two levels of operating environment: Single Job operation, and a powerful Foreground/Background capability. All the work described in this thesis has been performed under the Foreground/Background monitor, which is considered to have several advantages for this particular application. With this facility, it is possible to utilise the spare time which the central processor of the computer may have while waiting for an external event to occur. The monitor is arranged such that this event, and its associated program, is staged at high priority but any spare time may be used for lower priority tasks, such as further program development or complex data analysis tasks.

The monitor system itself performs a two-fold function. Firstly, it provides a number of basic program development aids which include an editor, assembler, linker, debugger and librarian. Secondly, it provides a supervised environment in which any real-time (or other) user programs may be run. This environment also provides a collection of programmed requests and utility functions to perform standard operations. The provision of

device handlers and associated buffer management is also one of the functions of the monitor system which facilitate easier and more flexible user programming.

13.2 DEVICE HANDLERS

13.2.1 Microprocessor Loader 'ML'

The device handler identified by the mnemonic 'ML' is a handler developed for loading program and/or data information to the I8Ø8Ø microprocessor through the hardware facilities of the Interface Card as described in Section 2. It has been specifically developed for use with microprocessor cross-assembler. 'MICRO', but functions equally well with any system or user program conforming to the standard configuration for device independent usage[45].

The handler accepts data for transmission to the microprocessor in the form of 16-bit words, which should be in the form of a control byte (high order) and a data byte (low order). This format is specified in Section 2. The handler detects when a control byte is a 'write-to-memory' command and subsequently performs an automatic 'read-from-memory' operation to confirm the data written. An error flag is generated if the data stored does not conform with the original request. Further operational details are given in the source program listing.

13.2.2 Microprocessor Data Output Handler 'MO'

This device handler, identified by the mnemonic 'MO', is for output of data to the microprocessor in a program-controlled mode, assuming the microprocessor has an input device handler or subroutine capable of receiving the data. It has been specifically developed for transfer of control data to the microprocessor during the real-time control operation, but functions equally well with any system or user program conforming to the requirements of device independent usage.

The handler accepts data for transmission to the microprocessor

as 8-bit bytes which it automatically structures into a formatted binary mode prior to transmission (refer to Section 7.4.3 of ref 70 for details). Thus, a check may be made by the device handler or subroutine in the microprocessor for the validity of the data received. The transfers are made on an interrupt basis, utilising hardware 'hand-shaking' facilities, for speed and efficiency of operation. Program returns are made through a completion interrupt subroutine to avoid program redundancy by waiting. Further operational details are given in the source program listing.

13.2.3 Microprocessor Data Input Handler 'MI'

Data input to the PDP-11 from the microprocessor may be handled by the device handler identified by the mnemonic 'MI'. This handler has been specifically developed for receiving data concerning the controlled system from the microprocessor during the real-time control operation. However, it will function equally well with any system or user program conforming to the configuration for device independent usage.

The data is accepted from the microprocessor, via the Interface Card, as 16-bit words in formatted binary mode (see Section 7.4.3 of ref 70 for details). The validity of data is checked by means of the checksum byte received and an error-free return to the calling program is only made if no bit errors are detected. All transfers are made on an interrupt basis, with completion returns to the calling sequence, to avoid program waiting redundancy. Further operational details are given in the source program listing.

13.3 RT-11 MONITOR MODIFICATIONS

The device handlers are incorporated into the RT-11 Monitor System by the following process, it is assumed that the source modules of the device handlers are available on magnetic tape as MOVØ2.MAC, MIVØ2.MAC and MLVØ2.MAC. The process described also serves to example other modifications to system and user

programs which are not detailed in the appendix.

13.3.1 Device Handler Assembly

The source modules are assembled from magnetic tape using the MACRO assembler[45]:

```
.DAT 9-FEB-77
.MACRO
*MO=MTØ:MOVØ2                    (Computer output
*MI=MTØ:MIVØ2                     is underlined)
*ML=MTØ:MLVØ2
*↑C
.
```

13.3.2 Linking

The handlers must be linked at 1ØØØ (default link address). It is important that '.SYS' is specified:

```
.R LINK
*MO.SYS=MO
*MI.SYS=MI
*ML.SYS=ML
*↑C
.
```

13.3.3 Modify the Monitor Handler Tables

The system device handler tables must be modified. The CT:, MM: and DP: handlers will be replaced such that Table 3 of the System Release Notes[71] will have the following (changed) entries:

Entry No	Octal Offset	Device Name ASCII	Device Name RAD5Ø	Device Code	Handler Size (Bytes)	Device Size	Status Word
10	22	MO	5163Ø	24	324	Ø	2ØØ24
11	24	ML	5144Ø	23	212	Ø	2ØØ23
13	3Ø	MI	5125Ø	25	3Ø2	Ø	4ØØ25

The patch is made as follows:

```
.R PATCH
PATCH VØ1-Ø2
FILE NAME--
*MONITR.SYS/M
*2ØØØØ;ØR
*Ø,146ØØ/     371Ø          324
*Ø,14634/     Ø             Ø
*Ø,14652/     1274Ø         5163Ø     } Patch for MO
*Ø,177Ø6/     12Ø13         2ØØ24
*Ø,17342/     Ø             14ØØØØ
*Ø,146Ø2/     47ØØ          212
*Ø,14636/     Ø             Ø
*Ø,17654/     5151Ø         5144Ø     } Patch for ML
*Ø,1771Ø/     12Ø2Ø         2ØØ23
*Ø,17342/     14ØØØØ        14ØØØØ
*Ø,146Ø6/     434           3Ø2
*Ø,14642/     11611Ø        Ø
*Ø,1766Ø/     156ØØ         5125Ø     } Patch for MI
*Ø,17714/     1ØØØ21        4ØØ25
*Ø,17342/     14ØØØØ        17ØØØØ
*E
.
```

13.3.4 Copy the Bootstrap

The bootstrap must be copied and the monitor re-booted:

```
.R PIP
*SY:A=MONITR.SYS/U
*SY:/O
RT-11FB  VØ2C-Ø2
.
```

SECTION 14

MICRO ASSEMBLER

MICRO is a 2-pass cross-assembler running on the PDP-11/20 computer and assembling machine code instructions for the Intel I8Ø8Ø microprocessor. Version one (latest update VØØ1-C) of the assembler was configured to run under the PDP-11 Disk Operating System and is no longer supported. The current version (VØØ2-A) is designed to run under the real-time RT-11 system and requires an RT-11 system configuration (or background partition) of 8k or more. Unless otherwise indicated, the following description applies to version VØØ2-A and it is assumed that the programmer is conversant with the major features of an assembler programme. Inexperienced programmers should refer first to the I8Ø8Ø Assembly Language Programming Manual[72] bearing in mind the following major differences of the MICRO cross-assembler:

1. All input/output conforms with the requirements of an RT-11 system program.
2. Macro definitions and expansions are not supported in the current version.
3. Conditional assembly facility not supported in the current version.
4. Some argument expressions are not supported.

Facility has been made for these features to be incorporated in later versions of the cross-assembler (see Section 14.8).

14.1 SOURCE PROGRAM FORMAT

A source program is composed of a sequence of source lines; each source line contains a single assembly language statement followed by a statement terminator. A terminator may be either a line feed character (which increments the line count by 1) or a form feed character (which resets the line count and increments the page count by 1).

NOTE: EDIT automatically appends a line feed to every carriage return encountered in a source program. For listing format, MICRO automatically inserts a carriage return before any line feed or form feed not preceded by one.

An assembly language line can contain up to 92 (decimal) characters (exclusive of the statement terminator). Beyond this limit, excess characters are ignored and generate an error flag.

14.1.1 Statement Format

A statement can contain up to four fields which are identified by order of appearance and by specified terminating characters. The general format of MICRO assembly language statement is:

label: operator operand(s) ; comments

The label and comment fields are optional. The operator and operand fields are independent; either may be omitted depending upon the contents of the other.

The assembler interprets and processes these statements one by one,

generating one or more binary instructions or data words or performing an assembly process. A statement contains one of these fields and may contain all four types. Blank lines are legal. An assembly language statement must be complete on one source line. No continuation lines are allowed. (If a continuation is attempted with a line feed, the assembler interprets this as the statement terminator).

MICRO source statements may be formatted with EDIT[45] so that the use of the TAB character causes the statement fields to be aligned. For example:

```
Label       Operator      Operand(s)          Comments

CHECK:      ANI           1Q                  ;IS THE NUMBER ODD?
            JZ            EVEN                ;NO, IT'S EVEN
            MVI           B,377Q              ;ELSE SET FLAG
EVEN:       RET                               ;RETURN
```

14.1.1.1 Label Field

A label is a user-defined symbol that is unique within the first six characters and is assigned the value of the current location counter and entered into the user-defined part of the symbol table. A label is a symbolic means of refering to a specific location within a program. If present, a label always occurs first in a statement and must be terminated by a colon. For example, if the current location is 1ØØ (octal), the statement

```
ABCD:     MOV     A,B
```

assigns the value 1ØØ (octal) to the label ABCD. Subsequent reference to ABCD references location 1ØØ (octal) or uses the value 1ØØ (octal) depending on context.

More than one label may appear within a label field, in which case each label within the field is assigned the same value. A symbol used as a label may not be redefined within the user program. An attempt to redefine a label results in an error flag in the assembly listing.

14.1.1.2 Operator Field

An operator field follows the label field in a statement and may contain an instruction mnemonic or an assembler directive. The operator may be preceded by zero, one or more labels and may be followed by one or more operands and/or a comment. Leading and trailing spaces and tabs are ignored.

When the operator is an instruction mnemonic, it specifies the instruction to be generated and the action to be performed on any operand(s) which follow. When the operator is an assembly directive, it specifies a certain function or action to be performed during assembly.

An operator is legally terminated by a space, tab, semi-colon (when there are no operands) or record terminator (when there are no operands or comments).

14.1.1.3 Operand Field

An operand is that part of the statement which is manipulated by the operator. Operands may be numbers or symbols (within the context of the operation). When two operands appear within a statement they must be separated by a comma. Spaces or tabs preceding or following the comma are ignored. An operand must be preceded by an operator and the statement may contain another

operand and a label and be followed by a comment.

The operand field is terminated by a semi-colon when followed by a comment, or by a statement terminator when the operand completes the statement.

14.1.1.4 Comment Field

The comment field is optional and may contain any ASCII characters except null, rubout, carriage return, line feed, vertical tab or form feed. All other characters, even special characters with defined usage, are ignored by the assembler when appearing in the comment field.

The comment field may be preceded by one, any, none or all of the other three field types. Comments must begin with the semi-colon character and end with a statement terminator. Comments do not affect assembly processing or program execution but are useful in source listings for later analysis, debugging or documentation purposes.

14.1.2 Format Control

Horizontal or line formatting of the source program is controlled by the space or tab characters. Inserting additional space or tab characters between fields has no effect on the assembly process and can thus be used to provide an orderly source program.

Vertical formatting, i.e. page size, is controlled by the form feed character. A page of n lines is created by inserting a form feed (CRTL FORM) after the nth line. (See also section 14.6 for a description of assembly listing output.

14.2 SYMBOLS AND EXPRESSIONS

This section describes the various components of legal MICRO expressions: the assembler character set, symbol construction and numbers.

14.2.1 Character Set

The following characters are legal in MICRO source programs:

1. The letters A to Z in upper case
2. The digits Ø to 9
3. The characters . (period or dot) and $ (dollar sign) which ' are reserved for use in system program symbols.
4. The following special characters:

Character	Designation	Function
carriage return		formatting character
line feed		source statement terminator
form feed		source statement terminator
vertical tab		source statement terminator
:	colon	label terminator
tab		item or field terminator
space		item or field terminator
(	left parenthesis	see note below
)	right parenthesis	" " "
,	comma	operand field separator
;	semi-colon	comment field indicator

Character	Designation	Function
+	plus sign	see note below
-	minus sign	" " "
*	asterisk	" " "
/	slash	" " "
'	single quote	" " "

Note These characters have been included in the legal character set to facilitate future expansion of MICRO to handle argument expressions.

A character can be illegal in one of two ways:

1. A character which is not recognised as an element of the MICRO character set is always an illegal character and causes immediate termination of the current line at that point, plus the output of an error flag in the assembly listing (I error).
2. A legal MICRO character may be illegal in context. Such a character will generate a Q error on the assembly listing.

14.2.2 Symbols

In the current version of MICRO there are two types of symbol: permanent and user-defined. However, no distinction is made between these symbols other than the permanent symbols are inserted into the symbol table by MICRO each time a new source listing is presented for assembly.

14.2.2.1 Permanent Symbols

Permanent symbols consist of instruction mnemonics, assembler directions and register mnemonics. These symbols are a permanent part of the assembler and need not be defined before being used in the source program. Only the register mnemonics, as given in Table 14.1, are entered into the User Symbol Table each time an assembly begins.

Table 14.1 Register Mnemonics

Mnemonic	Definition	Symbol Value
B	Register B or Register Pair B-C	Ø
C	Register C	1
D	Register D or Register Pair D-E	2
E	Register E	3
H	Register H or Register Pair H-L	4
L	Register L	5
M	Memory	6
A	Register A	7
SP	Stack Pointer	6
PSW	Register A and Condition Flags	6

Instruction mnemonics are listed in Table 14.2 and their operation explained in greater detail in the I8Ø8Ø Reference Manual[67]. Assembler directives are detailed in Section 14.3.

14.2.2.2 User-defined Symbols

User-defined symbols are those used as labels or defined by direct assignment (Section 14.3.4). These symbols are added to the User

Summary of Processor Instructions

Mnemonic	Description	Instruction Code[1] D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0	Clock[2] Cycles
MOV r1,r2	Move register to register	0	1	D	D	D	S	S	S	5
MOV M, r	Move register to memory	0	1	1	1	0	S	S	S	7
MOV r, M	Move memory to register	0	1	D	D	D	1	1	0	7
HLT	Halt	0	1	1	1	0	1	1	0	7
MVI r	Move immediate register	0	0	D	D	D	1	1	0	7
MVI M	Move immediate memory	0	0	1	1	0	1	1	0	10
INR r	Increment register	0	0	D	D	D	1	0	0	5
DCR r	Decrement register	0	0	D	D	D	1	0	1	5
INR M	Increment memory	0	0	1	1	0	1	0	0	10
DCR M	Decrement memory	0	0	1	1	0	1	0	1	10
ADD r	Add register to A	1	0	0	0	0	S	S	S	4
ADC r	Add register to A with carry	1	0	0	0	1	S	S	S	4
SUB r	Subtract register from A	1	0	0	1	0	S	S	S	4
SBB r	Subtract register from A with borrow	1	0	0	1	1	S	S	S	4
ANA r	And register with A	1	0	1	0	0	S	S	S	4
XRA r	Exclusive Or register with A	1	0	1	0	1	S	S	S	4
ORA r	Or register with A	1	0	1	1	0	S	S	S	4
CMP r	Compare register with A	1	0	1	1	1	S	S	S	4
ADD M	Add memory to A	1	0	0	0	0	1	1	0	7
ADC M	Add memory to A with carry	1	0	0	0	1	1	1	0	7
SUB M	Subtract memory from A	1	0	0	1	0	1	1	0	7
SBB M	Subtract memory from A with borrow	1	0	0	1	1	1	1	0	7
ANA M	And memory with A	1	0	1	0	0	1	1	0	7
XRA M	Exclusive Or memory with A	1	0	1	0	1	1	1	0	7
ORA M	Or memory with A	1	0	1	1	0	1	1	0	7
CMP M	Compare memory with A	1	0	1	1	1	1	1	0	7
ADI	Add immediate to A	1	1	0	0	0	1	1	0	7
ACI	Add immediate to A with carry	1	1	0	0	1	1	1	0	7
SUI	Subtract immediate from A	1	1	0	1	0	1	1	0	7
SBI	Subtract immediate from A with borrow	1	1	0	1	1	1	1	0	7
ANI	And immediate with A	1	1	1	0	0	1	1	0	7
XRI	Exclusive Or immediate with A	1	1	1	0	1	1	1	0	7
ORI	Or immediate with A	1	1	1	1	0	1	1	0	7
CPI	Compare immediate with A	1	1	1	1	1	1	1	0	7
RLC	Rotate A left	0	0	0	0	0	1	1	1	4
RRC	Rotate A right	0	0	0	0	1	1	1	1	4
RAL	Rotate A left through carry	0	0	0	1	0	1	1	1	4
RAR	Rotate A right through carry	0	0	0	1	1	1	1	1	4
JMP	Jump unconditional	1	1	0	0	0	0	1	1	10
JC	Jump on carry	1	1	0	1	1	0	1	0	10
JNC	Jump on no carry	1	1	0	1	0	0	1	0	10
JZ	Jump on zero	1	1	0	0	1	0	1	0	10
JNZ	Jump on no zero	1	1	0	0	0	0	1	0	10
JP	Jump on positive	1	1	1	1	0	0	1	0	10
JM	Jump on minus	1	1	1	1	1	0	1	0	10
JPE	Jump on parity even	1	1	1	0	1	0	1	0	10
JPO	Jump on parity odd	1	1	1	0	0	0	1	0	10
CALL	Call unconditional	1	1	0	0	1	1	0	1	17
CC	Call on carry	1	1	0	1	1	1	0	0	11/17
CNC	Call on no carry	1	1	0	1	0	1	0	0	11/17
CZ	Call on zero	1	1	0	0	1	1	0	0	11/17
CNZ	Call on no zero	1	1	0	0	0	1	0	0	11/17
CP	Call on positive	1	1	1	1	0	1	0	0	11/17
CM	Call on minus	1	1	1	1	1	1	0	0	11/17
CPE	Call on parity even	1	1	1	0	1	1	0	0	11/17
CPO	Call on parity odd	1	1	1	0	0	1	0	0	11/17
RET	Return	1	1	0	0	1	0	0	1	10
RC	Return on carry	1	1	0	1	1	0	0	0	5/11
RNC	Return on no carry	1	1	0	1	0	0	0	0	5/11
RZ	Return on zero	1	1	0	0	1	0	0	0	5/11
RNZ	Return on no zero	1	1	0	0	0	0	0	0	5/11
RP	Return on positive	1	1	1	1	0	0	0	0	5/11
RM	Return on minus	1	1	1	1	1	0	0	0	5/11
RPE	Return on parity even	1	1	1	0	1	0	0	0	5/11
RPO	Return on parity odd	1	1	1	0	0	0	0	0	5/11
RST	Restart	1	1	A	A	A	1	1	1	11
IN	Input	1	1	0	1	1	0	1	1	10
OUT	Output	1	1	0	1	0	0	1	1	10
LXI B	Load immediate register Pair B & C	0	0	0	0	0	0	0	1	10
LXI D	Load immediate register Pair D & E	0	0	0	1	0	0	0	1	10
LXI H	Load immediate register Pair H & L	0	0	1	0	0	0	0	1	10
LXI SP	Load immediate stack pointer	0	0	1	1	0	0	0	1	10
PUSH B	Push register Pair B & C on stack	1	1	0	0	0	1	0	1	11
PUSH D	Push register Pair D & E on stack	1	1	0	1	0	1	0	1	11
PUSH H	Push register Pair H & L on stack	1	1	1	0	0	1	0	1	11
PUSH PSW	Push A and Flags on stack	1	1	1	1	0	1	0	1	11
POP B	Pop register pair B & C off stack	1	1	0	0	0	0	0	1	10
POP D	Pop register pair D & E off stack	1	1	0	1	0	0	0	1	10
POP H	Pop register pair H & L off stack	1	1	1	0	0	0	0	1	10
POP PSW	Pop A and Flags off stack	1	1	1	1	0	0	0	1	10
STA	Store A direct	0	0	1	1	0	0	1	0	13
LDA	Load A direct	0	0	1	1	1	0	1	0	13
XCHG	Exchange D & E, H & L Registers	1	1	1	0	1	0	1	1	4
XTHL	Exchange top of stack, H & L	1	1	1	0	0	0	1	1	18
SPHL	H & L to stack pointer	1	1	1	1	1	0	0	1	5
PCHL	H & L to program counter	1	1	1	0	1	0	0	1	5
DAD B	Add B & C to H & L	0	0	0	0	1	0	0	1	10
DAD D	Add D & E to H & L	0	0	0	1	1	0	0	1	10
DAD H	Add H & L to H & L	0	0	1	0	1	0	0	1	10
DAD SP	Add stack pointer to H & L	0	0	1	1	1	0	0	1	10
STAX B	Store A indirect	0	0	0	0	0	0	1	0	7
STAX D	Store A indirect	0	0	0	1	0	0	1	0	7
LDAX B	Load A indirect	0	0	0	0	1	0	1	0	7
LDAX D	Load A indirect	0	0	0	1	1	0	1	0	7
INX B	Increment B & C registers	0	0	0	0	0	0	1	1	5
INX D	Increment D & E registers	0	0	0	1	0	0	1	1	5
INX H	Increment H & L registers	0	0	1	0	0	0	1	1	5
INX SP	Increment stack pointer	0	0	1	1	0	0	1	1	5
DCX B	Decrement B & C	0	0	0	0	1	0	1	1	5
DCX D	Decrement D & E	0	0	0	1	1	0	1	1	5
DCX H	Decrement H & L	0	0	1	0	1	0	1	1	5
DCX SP	Decrement stack pointer	0	0	1	1	1	0	1	1	5
CMA	Compliment A	0	0	1	0	1	1	1	1	4
STC	Set carry	0	0	1	1	0	1	1	1	4
CMC	Compliment carry	0	0	1	1	1	1	1	1	4
DAA	Decimal adjust A	0	0	1	0	0	1	1	1	4
SHLD	Store H & L direct	0	0	1	0	0	0	1	0	16
LHLD	Load H & L direct	0	0	1	0	1	0	1	0	16
EI	Enable Interrupts	1	1	1	1	1	0	1	1	4
DI	Disable Interrupt	1	1	1	1	0	0	1	1	4
NOP	No-operation	0	0	0	0	0	0	0	0	4

NOTES: 1. DDD or SSS – 000 B – 001 C – 010 D – 011 E – 100 H – 101 L – 110 Memory – 111 A.
2. Two possible cycle times, (5/11) indicate instruction cycles dependent on condition flags.

Table 14.2 I8Ø8Ø Instruction Set (Mnemonics)

Symbol Table as they are encountered during the first pass of the assembly. User-defined symbols can be composed of alphanumeric characters, dollar signs and periods only; any other character is illegal. The '$' and '.' characters are generally reserved for system software symbols and it is recommended that they are not inserted in user-defined symbols. The following rules apply to the creation of user-defined symbols:

1. The first character must not be a number
2. Each symbol must be unique within the first six characters
3. A symbol can be written more than six characters, but the seventh and subsequent characters are only checked for legality, and are not otherwise recognised by the assembler.
4. Spaces, tabs and illegal characters must not be embedded within a symbol.

In the current version of the assembler all user-defined symbols are assigned absolute values i.e. no allowance is made for program relocation. If relocation is desired than, generally, the source must be re-assembled with a new origin specified (Section 14.3.3.2).

14.2.3 Direct Assignment

A direct assignment statement allocates a symbol with a value and makes the relevant entry in the User Symbol Table. This is performed by the SET or EQU assembly directives as detailed in Section 14.3.4.

14.2.4 Numbers

The MICRO assembler assumes no default radices for numbers specified in the source program. All numbers must be specified as being one of three types: octal, decimal or hexadecimal. Depending on the operand, the number may be used to define one or two bytes of assembled program code. Generally, a truncation error (flagged as T in the listing) will occur if the converted number is too large to fit into eight or sixteen bits. The assembler works in two's complement arithmetic such that leading zeros or leading ones can be dropped without loss of numerical significance. For example, the sixteen bit number 177634_8 (= -100_{10}) can be expressed in eight bits as 234_8 (= -100_{10}) and will not be flagged by the assembler as an error when converted to fill a single byte.

14.2.4.1 Octal Numbers

Octal numbers consist of the digits'Ø' to '7' only and must be followed immediately by the letter 'O' or 'Q'. They will normally have a maximum value of 377Q when being used to define a byte of assembled code or 177777Q when being used to define two bytes of code (within the provisions of truncation as given in Section 14.2.4)

Valid octal byte definitions are, for example:

37Q	(= 33_{10})
3Ø4O	(= -60_{10})
234Q	(= -100_{10})
177634Q	(truncates to -100_{10} without loss of significance)

Invalid byte definitions are:

47	(No 'O' or 'Q' character)
28Q	(Contains digit '8')
17Ø43Q	(Too many significant bits)

Note: the last example would be valid for an operand defining two bytes of data

14.2.4.2 Decimal Numbers

Decimal numbers consist of the digits 'Ø' to '9' and must be followed immediately by the letter 'D'. They will normally have a maximum value of 255D when being used to define a byte of assembled code and 65535D when being used to define two bytes of code.

Valid decimal byte definitions are, for example:

25D	(= $25_{1Ø}$)
173D	(= $173_{1Ø}$)

Invalid byte definitions are:

46	(No 'D' character)
427D	(Too many significant bits)

Note: the last example would be valid for an operand defining two bytes of data

14.2.4.3 Hexadecimal Numbers

Hexadecimal numbers consist of the digits 'Ø' to '9' and the letters 'A' to 'F' and followed by a letter 'H'. They must begin with a numeric digit. They will normally have a maximum value of

ØFFH when being used to define a byte of assembled code and ØFFFFH when being used to define two bytes of code.

Valid hexadecimal byte definitions are, for example:

ØBAH	($= -7Ø_{1Ø}$)
ØFF3H	(truncates to $-3_{1Ø}$ without loss of significance)

Invalid byte definitions are:

7A	(No 'H' character)
F3H	(First letter not numeric)
ØCB7H	(Too many significant bits)

Note: The last example would be valid for an operand defining <u>two</u> bytes of assembled data

14.3 ASSEMBLER DIRECTIVES

Directives are statements which cause the assembler to perform certain processing operations.

Assembler directives can be preceded by a label, subject to restrictions associated with specific directives, and followed by a comment. An assembler directive occupies the operator field of a MICRO source line. Only one directive can be placed on any one line. Zero, one or more operands can occupy the operand field; legal operands differing with each directive but may, in general, be symbols or numbers.

14.3.1. Listing Directives

14.3.1.1 Page Headings

The MICRO Assembler outputs each page in the format shown in Figure 14.1. On the first line of each listing page the assembler prints (from left to right)

1. Title taken from TITLE directive (most recent one encountered)
2. Assembler version identification
3. The date and time of day if entered
4. The page number

14.3.1.2 TITLE Directive

The title directive is used to print a heading in the output listing. The heading printed on the first line of each page of the listing is taken from the argument in the TITLE directive which should be such that not more than 9Ø printing characters in the listing will be used (after tabs have been expanded).

If there is no TITLE statement, the default name used as a titles is'.MAIN.' . The first tab or space following the TITLE directive is not considered part of the header text, although subsequent spaces and tabs are significant. If there is more than one TITLE directive, the last TITLE directive in the program conveys the program name.

14.3.1.3 Page Ejection

There are two means of obtaining a page ejection in a MICRO assembly

Fig. 14.1 'MICRO' Cross Assembler Listing Example

```
CYCTST                                                    MICRO V002-A  16-SEP-77  00:00.00  PAGE  1

                                ;
                                ;       P. J. BURROWS                           UNIVERSITY OF BATH
                                ;
                                ;       PROGRAM TO RUN A CYCLIC DATA TEST ON THE PDP-11/20 TO I8080
                                ;       TRANSMISSION LINK
                                ;
                                        TITLE   CYCTST
                                        ORG     00
000000      303     100     000         JMP     MAIN                    ;AT START-UP JUMP TO MAIN PROGRAM
                                        ORG     1000
000100      041     147     000 MAIN:   LXI     H,DATA                  ;SET POINTER TO DATA PATTERN
000103      116                 LOOP1:  MOV     C,M
000104      043                         INX     H
000105      106                         MOV     B,M                     ;PATTERN NOW IN REGISTER PAIR B-C
000106      026     010         LOOP1:  MVI     D,8D                    ;MAJOR CYCLE COUNT
000110      036     377         LOOP2:  MVI     E,0FFH                  ;MINOR CYCLE COUNT
000112      035                 LOOP3:  DCR     E
000113      302     112     000         JNZ     LOOP3                   ;MINOR LOOP
000116      025                         DCR     D
000117      302     110     000         JNZ     LOOP2                   ;MAJOR LOOP
000122      067                         STC
000123      077                         CMC
000124      171                         MOV     A,C
000125      027                         RAL
000126      117                         MOV     C,A
000127      170                         MOV     A,B                     ;ROTATE PATTERN LEFT
000130      027                         RAL
000131      107                         MOV     B,A
000132      171                         MOV     A,C
000133      316     000                 ACI     00                      ;SEND DOWN LAST BIT
000135      117                         MOV     C,A
000136      170                         MOV     A,B
000137      323     377                 OUT     377Q                    ;DISPLAY HIGH ORDER
000141      171                         MOV     A,C
000142      323     000                 OUT     0H                      ;DISPLAY LOW ORDER
000144      303     106     000         JMP     LOOP1                   ;REPEAT CYCLE
000147      003     000         DATA:   DW      3D                      ;PATTERN = 0000000000000011
                                        END

CYCTST                                                    MICRO V002-A  16-SEP-77  00:00.00  PAGE  2

B       000000          C       000001          D       000002          E       000003          H       000004
L       000005          M       000006          A       000007          SP      000006          PSW     000006
MAIN    000100          LOOP1   000106          LOOP2   000110          LOOP3   000112          DATA    000147

ERRORS DETECTED: 0
FREE CORE: 10906. WORDS
```

listing:

1. After a line count of 61 lines, MICRO automatically performs a page eject to skip over page perforations on line printer paper and to formulate terminal output into pages.
2. A form feed (CRTL FORM) character used as a line terminator (or as the only character on a line) causes a page eject.

14.3.2 Data Storage Directives

In the current version of the assembler data may only be stored by one of two directives, the Define Byte and Define Word directives.

14.3.2.1 DB Directive

The DB (Define Byte) directive is used to generate successive bytes of data. It may optionally have a label and/or comment and is of the form:

```
Label:      DB      arg         ;comment
```

The argument may be a valid byte number as defined in Section 14.2.4. or a previously-defined symbol in the User Symbol Table.

14.3.2.2 DW Directive

The DW (Define Word) directive is used to generate two bytes of data from a single argument. It may optionally have a label and/or comment and is of the form:

```
Label:      DW      arg         ;comment
```

The argument may be a valid two-byte number as defined in section 14.2.4 or a previously-defined symbol in the User Symbol Table.

14.3.3 Location Counter Control

14.3.3.1 Automatic Facility

The assembly location counter governs the address in microprocessor memory at which the current instruction will be assembled. It is automatically set to zero at the beginning of the assembly process and is incremented by 1,2 or 3 following the assembly of one- , two- or three-byte instructions.

14.3.3.2 ORG Directive

The location at which the following program code will be assembled can be changed by the ORG directive. It can have optional label and/or comment fields and is of the following form:

```
label:      ORG      arg          ;comment
```

The optional label will have a value one greater than the address of the last byte assembled, not the value of the argument. The argument may be a valid 16-bit number as defined in Section or a previously-defined symbol in the User Symbol Table.

The ORG directive can be used to reserve areas of memory for data storage but it should be noted that these areas will not contain assembled data. In particular, the programmer should not assume that these areas contain zero, or any other value.

14.3.4 Direct Assignments

A direct assignment statement allocates a symbol with a value and makes the relevant entry in the User Symbol Table.

14.3.4.1 EQU Directive

The EQU (Equate) directive assigns a value to a symbol name and whenever that symbol is encountered subsequently in the assembly, the value will be used. The statement may have an optional label and/or comment field but the symbol name must appear in the label field without a following colon:

```
name        EQU         arg             ;comment
```

The symbol 'name' and 'arg' are both necessary and the symbol may appear in the name field of only one EQU directive, i.e. an EQU symbol may not be redefined.

14.3.4.2 SET directive

The SET directive assigns a value to a symbol name and whenever that symbol is encountered subsequently in the assembly, the value will be used. The statement may have an optional label and/or comment field but the symbol name must appear in the label field without a following colon:

```
name        SET         arg             ;comment
```

The symbol 'name' and 'arg' are both necessary but the symbol name may be redefined by a later SET directive. This is identical with the EQU directive except the symbol may be defined more than once.

14.3.5Terminating Directives

14.3.5.1 END Directive

The END directive indicates the physical end of the source program. It may have an optional label and/or comment field but supports no argument. Any statements following an END directive will be ignored by the assembler

14.3.5.2 Physical End of File

If the assembler attempts to read past the physical end-of-file of the input program before an END directive is reached then it will generate an E error flag and assume that such a directive has been found.

14.4 CALLING AND USING MICRO

The MICRO assembler assembles one ASCII source file containing MICRO statements into a single absolute binary load file. Assembler output consists of this binary load file and an optional assembly listing followed by the symbol table listing.

MICRO is executed using the RT-11 Monitor R command[45] as follows:

```
.R MICRO
```

The assembler responds by typing an asterisk (*) to indicate readiness to accept command string input. In response to the * printed by the assembler, the user types the output file specification(s), followed by an equal sign, followed by the input file specification as follows:

```
*dev:load, dev:list = dev:source/s
```

where

dev:	is any legal RT-11 device for output; must be file structured for input
load	is the binary load file
list	is the assembly listing file containing the assembly listing and symbol table
source	is the ASCII source file containing the MICRO source program
/s	represents an optional switch as detailed in

A null specification in either of the output file fields signifies that the associated output file is not desired. The default case for each file specification is noted below:-

file	device	filename	extension
load	DK:	-	.MLM
list	DK:	-	.LST
source	DK:	-	.MIC

Type CRTL C at any time to halt MICRO and return control to the monitor. To restart the assembler type R MICRO or the REENTER command in response to the monitor's dot.

14.4.1 Switches

In the current version of the assembler, only one switch option is available and this is the /I switch which identifies the version of the assembler in use. It is used as an input file switch and thus may be used to check the assembler version without specifying any files as follows:

```
. R MICRO
* /I                          (Computer output is
                                underlined)
MICRO VØØ2-A
*
```

Another command string may then be entered. Alternatively, the /I switch may be used as part of a normal command string.

14.5 MICRO ERROR MESSAGES

MICRO error messages enclosed in question marks are output on the terminal. The single letter error codes are printed in the assembly listing

Error Code	Meaning
E	END directive not found (An END is generated)
I	Illegal character detected. Depending on the context of the illegality (Section 14.2.1) the character may be ignored.
L	Line buffer overflow, i.e. input line greater than 92 characters. Extra characters on line are ignored.
P	Phase error. A label's definition or value varies from one pass to another.
Q	Questionable syntax. There are missing arguments or the instruction scan was not completed.
U	Undefined Symbol. An undefined symbol was encountered as the argument of a statement. The argument is assigned the value zero.
*	An assembler directive has been encountered which is not supported in this version of the assembler but which may be included in later versions (e.g. IF, ENDIF, MACRO, etc).

Error Message	Explanation
?BAD SWITCH?	The switch specified was not recognised by the program
?TOO MANY OUTPUT FILES?	Too many output files were specified
?NO INPUT FILE?	No input file was specified and there must be an input file (unless /I is used alone)
?TOO MANY INPUT FILES?	The current version of the assembler will handle only one input file
?INPUT ERROR?	A hardware error (other than end-of-file) occurred while trying to read from the source file
?WRITE ERROR?	A hardware error occurred while attempting to write to an output file
?INSUFFICIENT CORE?	There is insufficient free core to allow the program to run. A minimum area of 66_{10} free words is required for the User Symbol Table once the program and device handlers are in core
?SYMBOL TABLE FULL?	The program has exhausted all of the available symbol table space. This may be increased by using fewer device handlers (e.g. listing file on DK: and not LP:).
?INTERNAL FAULT?	The assembler is unable to process the source file even for flagging errors. The MICRO assembler requires modification. (This error message is only included to handle unforseen errors and should never occur in normal usage)

14.6 MICRO LISTING EXAMPLE

An example of the listing output of the MICRO cross-assembler is given in Figure 14.1. The first line of output on each page is (from left to right): the program title as given in a TITLE assembler directive (Section 14.3.1.2) followed by the assembler name and version number, the date and time of assembly and the page number.

Each subsequent line then contains information in one or more of the following fields depending on the contents of the corresponding input line

Printer Columns	Field
1 upwards	Error Codes - Refer to Section 14.5
1 to 7	Input Line Number - the line number of each line of the source program (in decimal)
9 to 14	Assembly Location Counter - the absolute location in microprocessor memory at which the following bytes will be loaded (in octal)
20 to 22 28 to 30 36 to 38	Assembled Bytes - One, two or three bytes which will be loaded into the microprocessor memory starting at the address specified in the previous field (in octal)
41 to 132	Input Statement - the input statement is listed as it appears in the source program to a maximum of 92 characters.

The final page of the assembly listing output contains the Symbol Table. Each of the symbols encountered in the assembly process is listed with its absolute value. In the case of re-defined symbols (from SET assembler directives), the last defined value is listed.

Finally, the assembler lists the number of errors encountered in the assembly process and also the free core available during assembly. This information is also given on the system terminal in case a listing output is not requested.

14.7 MICRO OUTPUT EXAMPLE

The binary output produced by the MICRO assembler from the example of Figure 14.1 is illustrated by Figure 14.2. The actual output is in pure binary form but a listing of this output in octal is given in the figure, obtained by means of the DUMP system program[45].

Each 16-bit word is composed of two parts: the high-order byte (8 bits) which serves as a control command to the PDP-11/20 to I8080 Interface Card (see Section 2) and the low-order byte which is utilised by the Interface Card in a manner dependent on the contents of the high-order byte.

Of the many functions performed by the Interface Card, the MICRO assembler utilises only the following (full details are given in Section 2):-

Output Word	Function
00100_8	HOLD Request. The microprocessor enters a

```
BLOCK NUMBER 00000
000/ 001000 001400 005000 105000 061303 061100 061000 005100
020/ 105000 061041 061147 061000 061116 061043 061106 061026
040/ 061010 061036 061377 061035 061302 061112 061000 061025
060/ 061302 061110 061000 061067 061077 061171 061027 061117
100/ 061170 061027 061107 061171 061316 061000 061117 061170
120/ 061323 061377 061171 061323 061000 061303 061106 061000
140/ 061003 061000 001000 001000 001000 001000 001000 001000
160/ 001000 001000 001000 001000 001000 001000 001000 001000
200/ 001000 001000 001000 001000 001000 001000 001000 001000
220/ 001000 001000 001000 001000 001000 001000 001000 001000
240/ 001000 001000 001000 001000 001000 001000 001000 001000
260/ 001000 001000 001000 001000 001000 001000 001000 001000
300/ 001000 001000 001000 001000 001000 001000 001000 001000
320/ 001000 001000 001000 001000 001000 001000 001000 001000
340/ 001000 001000 001000 001000 001000 001000 001000 001000
360/ 001000 001000 001000 001000 001000 001000 001000 001000
400/ 001000 001000 001000 001000 001000 001000 001000 001000
420/ 001000 001000 001000 001000 001000 001000 001000 001000
440/ 001000 001000 001000 001000 001000 001000 001000 001000
460/ 001000 001000 001000 001000 001000 001000 001000 001000
500/ 001000 001000 001000 001000 001000 001000 001000 001000
520/ 001000 001000 001000 001000 001000 001000 001000 001000
540/ 001000 001000 001000 001000 001000 001000 001000 001000
560/ 001000 001000 001000 001000 001000 001000 001000 001000
600/ 001000 001000 001000 001000 001000 001000 001000 001000
620/ 001000 001000 001000 001000 001000 001000 001000 001000
640/ 001000 001000 001000 001000 001000 001000 001000 001000
660/ 001000 001000 001000 001000 001000 001000 001000 001000
700/ 001000 001000 001000 001000 001000 001000 001000 001000
720/ 001000 001000 001000 001000 001000 001000 001000 001000
740/ 001000 001000 001000 001000 001000 001000 001000 001000
760/ 001000 001000 001000 001000 001000 001000 001000 001000
```

(listing in octal)

Fig. 14.2 'MICRO' Cross Assembler Output (Object File) Example

Output Word	Function
	'HOLD' state which allows the PDP-11/20 to have direct memory access (DMA) to the microprocessor memory and peripherals.
00140_8	HOLD and RESET. The microprocessor's internal program counter and instruction register are reset to zero. The Interface Card load register and transfer flags are also reset.
$005LLL_8$	HOLD and LOAD ADDR(LOW). The byte LLL is loaded into the low-order part of the Interface load register (i.e. re-specifies loading address of data following).
$105HHH_8$	HOLD and LOAD ADDR(HIGH). The byte HHH is loaded into the high-order part of the Interface load register (i.e. re-specifies loading address of data following).
$061DDD_8$	HOLD and WRITE to MEMORY (AUTO-INC). The byte DDD is written into the memory location currently referenced by the Interface load register. The load register is automatically incremented by one after the data has been written.

Thus, in the example of Figure 14.2, the following functions are performed (all numbers in octal):-

Word Address	Contents	Function
ØØØ	ØØ1ØØØ	HOLD Request
ØØ2	ØØ14ØØ	HOLD and RESET
ØØ4	ØØ5ØØØ	Reset load register to ØØØØØØ
ØØ6	1Ø5ØØØ	
Ø1Ø	Ø613Ø3	Load first three locations of
Ø12	Ø611ØØ	memory with 3Ø3, 1ØØ and ØØØ
Ø14	Ø61ØØØ	
Ø16	ØØ51ØØ	Reset load register to ØØØ1ØØ
Ø2Ø	1Ø5ØØØ	
Ø22	Ø61Ø41	Loads the bytes Ø41, 147 etc
Ø24	Ø61147	into memory location ØØØ1ØØ
etc	etc	upwards

14.8 FUTURE EXPANSIONS

Provision has been made in the MICRO cross-assembler for the later inclusion of additional assembly directives, argument expressions, etc, which time has not allowed the inclusion of in the current version. These are notably:

1. Conditional assembly (IF and ENDIF statements)
2. Macro definitions and expansions (MACRO and ENDM)
3. Argument expressions (eg 3+VALUE*2)

Reference should be made to the MICRO Assembler Source Program if it is desired to include these extra facilities. A much simplified flow-chart of the MICRO Assembler is given in Figure 14.3.

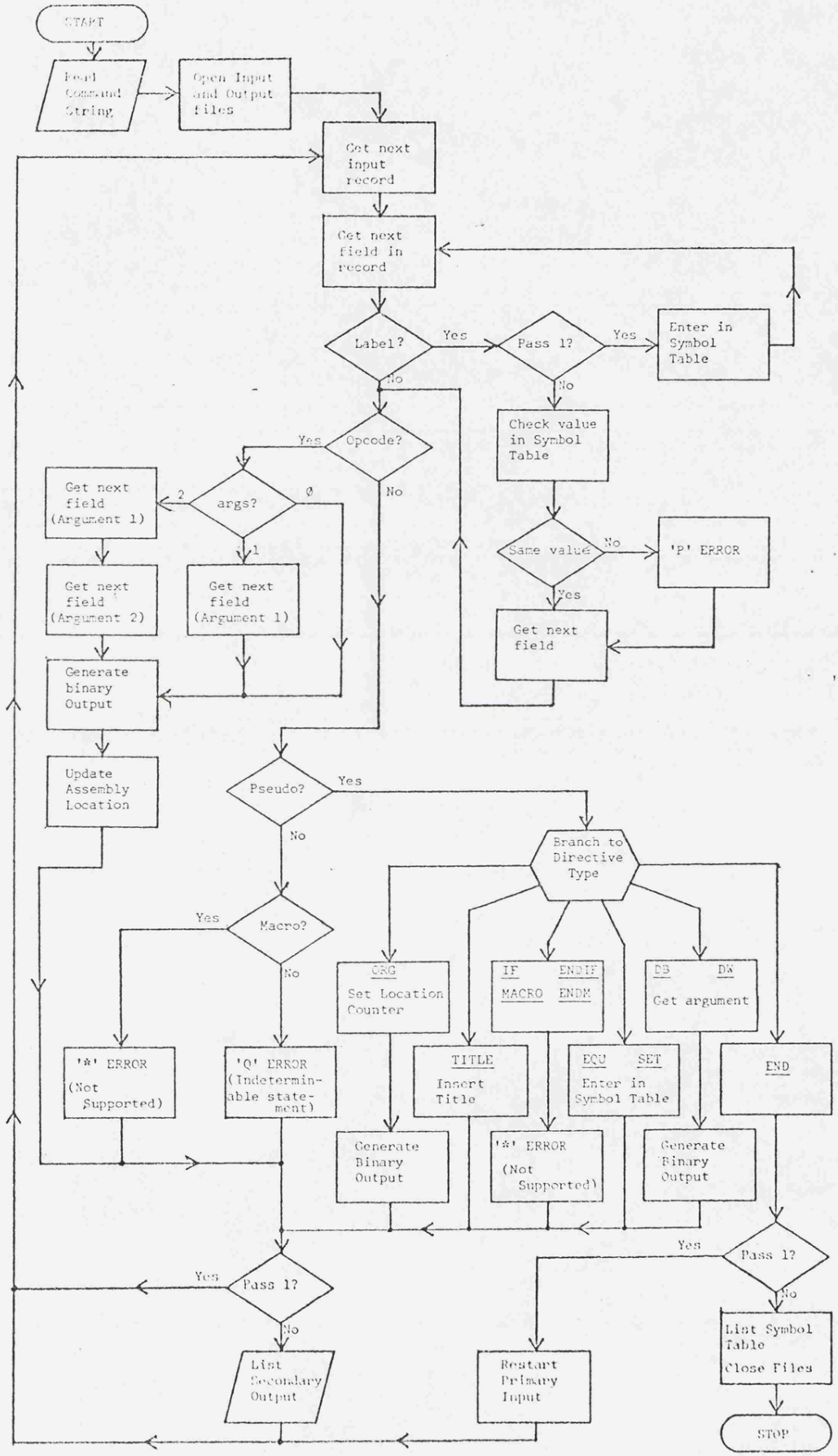


Fig. 14.3 'MICRO' Cross-Assembler Flow Diagram (Simplified)

SECTION 15

'DEBUG' ON-LINE DEBUGGING TECHNIQUE

DEBUG is an on-line system program that aids in debugging assembled microprocessor programs. From the keyboard of the PDP-11/2Ø the user interacts with DEBUG and the user-written program to perform various operations. These include:

1. Printing the contents of any memory location for examination or alteration.

2. Running all or a portion of an object program using the breakpoint feature.

3. Examining the microprocessor status, address and data buses to check program operation.

4. Directly opening a microprocessor input or output port for immediate data transfer.

5. Sending interrupts and data to the microprocessor via the interface and reading data from the microprocessor to simulate real-time data transfers.

The assembly listing of the program to be debugged should be readily available when DEBUG is being used. Minor corrections to the program can be made on-line during the debugging session, and the program may then be run, optionally under the control of DEBUG, to verify any changes made. Major corrections, however, should be noted on the assembly listing and incorporated in a subsequent updated program listing.

15.1 CALLING AND USING 'DEBUG'

DEBUG exists on the system disk as a linked memory image module (DEBUG.SAV). The current version (VØØ2-A) may be run as a fore-

ground or background program but may not be linked with any other image module, such as a user program. It is anticipated that this drawback may be overcome in later versions.

The user program to be debugged must be loaded into the microprocessor memory by one of the means available. Typically, using PIP[45]:

```
.R PIP                              (Computer output
*ML:=dev:fname.ext                    is underlined)
*
```

where dev: is any legal RT-11 device containing the load module (MICRO assembler output) of the program to be debugged. Default value is SY:

fname is the name of the load module file. No default value.

ext is the extension of the load module file. No default value.

The ML device handler has a read-back check on loading, thus a correct load is signified by no write errors being reported. The DEBUG program may then be run from the normal monitor command.

```
.R DEBUG
DEBUG VØØ2-A
*
```

DEBUG has two entry addresses; the normal entry (START) is made by the monitor R command and the re-entry address (START-2) is made by the monitor RE command. Both entries clear all breakpoints and data transfer registers. The microprocessor program may be re-started, retaining the breakpoints, as described in Section 15.2.5.1.

15.1.1 Recalling DEBUG command mode, CTRL D

If DEBUG is not in command mode, signified by the absence of a prompting asterisk on the teletype, then it will not respond to any normal DEBUG commands. This may occur in the running of a long program section or in a closed loop without breakpoints. The command mode may be re-entered by typing CTRL D. All other characters except CTRL C and CTRL O are ignored. On receipt of CTRL D, DEBUG echoes ↑D and ceases the current operation. The microprocessor enters a WAIT state at the current instruction and no breakpoint or data-transfer information is lost. DEBUG then types an asterisk to indicate its readiness to accept a new command.

15.1.2 Return to Monitor, CTRL C

If DEBUG is awaiting a command, a CTRL C from the keyboard calls the RT-11 Keyboard Monitor. The monitor responds with a ↑C on the terminal and awaits a Keyboard Monitor Command. The monitor REENTER command may be used as described above to clear the breakpoints etc. If DEBUG is not in command mode then two CTRL C characters from the keyboard are necessary to recall the Keyboard Monitor.

15.2 COMMANDS AND FUNCTIONS

When DEBUG is started it indicates its readiness to accept commands by printing an asterisk on the left margin of the terminal page. All DEBUG commands except CTRL D (section 15.1.1) are issued in response to this asterisk.

15.2.1 Number Formats

When accepting numbers from the teletype, DEBUG will normally accept them in one of three forms. These forms are:

1. Single numeric characters which specify a breakpoint (see section 15.2.4)

2. A string of numeric characters to represent a byte (8-bit) value.

3. A string of numeric characters to represent a two-byte (16-bit) value.

In case 1 no truncation is performed and a number outside the range $Ø_8$ to 7_8 is flagged as an error (see Section 15.3.2). In both other cases the numeric input string is interpreted as an octal number, i.e. only characters Ø to 7 are legal, and is truncated from the left to fill the data requirements of the specific instruction. No errors are given for any legal octal number. For example:

Number typed	DEBUG Interpretation	
	(Byte)	(2-Byte)
3	$ØØ3_8$	$ØØØØØ3_8$
17463	$Ø63_8$	$Ø17463_8$
3271361	361_8	$Ø71361_8$
ØØ7	$ØØ7_8$	$ØØØØØ7_8$

15.2.2 Opening, Changing and Closing Memory Locations

An open location is one whose contents DEBUG prints for examination, making those contents available for change. In a closed location, the contents are no longer available for change.

15.2.2.1 Opening a memory location

If no memory location is currently open then any valid location may be opened by typing its address followed by a slash character, e.g.:

```
* 4173 / 176          (Computer output is underlined)
```

Location 4173_8 is opened and its contents, 176_8, are available for change. If no address is specified then the last opened address is re-opened. This facility is useful for periodic checking of the same location. Additionally, memory locations may be opened as a result of closing preceding or following locations, see section 15.2.2.3.

15.2.2.2 Changing a memory location

Once open, the contents of a memory location may be changed by simply typing a valid octal number followed by one of the legitimate location closing characters. The modification to the memory location is not made until a legitimate closing character is detected. A character illegal in context will cause '?' to be echoed and the location will be closed without modification. Thus if a typing error is made the RUBOUT may be used to close the location without modification and a single slash will re-open it.

15.2.2.3 Closing a memory location

An open location may be closed legitimately by one of three characters, any of which may be preceded by a valid octal byte if it is desired to change the contents of the location. These three characters are:

1. Carriage return (represented <CR>). This closes the location and awaits the next command.

2. Line feed (represented <LF>). This closes the currently open location and opens the next memory location (higher address)

3. Vertical arrow (represented ↑). This closes the currently open location and opens the preceding memory location (lower address)

For example:

* 4173 / 176 2Ø2 <CR> (Change contents to $2Ø2_8$)

*/2Ø2 <LF> (Check contents and open next location)

*ØØ4174/333

15.2.3 Direct Access Input/Output

In a similar manner to memory location opening, DEBUG provides the facility for direct access of the input and output ports of the microprocessor system.

15.2.3.1 Opening and Reading from an Input Port

An input port may be opened by typing its port number followed by a right oblique parenthesis. DEBUG then reads the port and types its contents and awaits a legitimate port closing character (section 15.3.3.3) The contents may not be modified. For example:

*24>213 (Port 24_8 has data 213_8 for input)

It should be noted that this operation will cause a DATA TMTD signal to be sent to the peripheral connected to the port. If the port number is omitted then the last-opened input port is re-opened.

15.2.3.2 Opening and Writing to an Output Port

An output port may be opened by typing its port number followed by a left oblique parenthesis. A byte value may then be typed which DEBUG will transmit to the relevant port on detection of a valid port closing character. For example:

*16<Ø24 <CR> (sends $Ø24_8$ to port 16_8)

*

The data will not be transmitted until a legitimate closing character is detected thus enabling the use of the RUBOUT key. It should be noted that this output operation also sends a DATA

RDY signal to the peripheral unit connected to it. If the port number is omitted then the last-opened output port is re-opened.

15.2.3.3 Closing an Input or Output Port

An input or output port may be closed by one of three characters which have the functions detailed in section 15.2.2.3 except that the action is applied to input or output ports instead of memory locations. For example:

```
*2Ø>377↑                (port 2Ø8 has 3778 as data)
*Ø17>216↑               (port 178 has 2168 as data)
*Ø16>Ø14 <CR>           (port 168 has Ø148 as data)
*
```

15.2.4 Breakpoint Operation

The breakpoint facility allows sections of a program to be tested and program flow to be checked. A breakpoint trap may be set at any valid memory address and a program WAIT will be instigated when this address is referenced in the course of normal program flow.

15.2.4.1 Breakpoint Setting

DEBUG contains the facility to set up to eight breakpoints which are numbered from Ø to 7. A breakpoint may be set explicitly by typing an address followed by the breakpoint number. For example:-

```
*Ø1Ø2;3B
*
```

set breakpoint number 3 to the address ØØØ1Ø28 and will cause a WAIT when this address is reached in normal program flow. A breakpoint may also be set implicitly by typing an address followed by the letter 'B'. For example:

```
*3216;B
*
```

sets the next un-allocated breakpoint (starting from Ø) to the address $ØØ3216_8$.

15.2.4.2 Breakpoint Information

Whenever the DEBUG program waits at a breakpoint it types the breakpoint number followed by the address of the breakpoint. For example:

```
*;P
B3;ØØØ1Ø2
```

means that the microprocessor program has reached address ØØØ1Ø2 to which breakpoint no 3 has been set and is now WAITing at that location. If no breakpoint has been reached, for example in Single-Step mode or following an 'R' or 'W' instruction, then the same information is output with the default value of the (non-existent) breakpoint no 8 as follows:

```
*;W
B8;ØØØ372
*
```

signifying that the program is WAITing at location $ØØØ372_8$.

15.2.4.3 Clearing Breakpoints

A specific breakpoint may be cleared by specifying no address to the setting command. For example:

```
*;3B
*
```

clears breakpoint no 3. Similarly, all breakpoints may be cleared by the command:

```
*;B
*
```

All breakpoints may also be cleared by restarting the DEBUG program (section 15.1).

15.2.5 Program Running Instructions

15.2.5.1 Re-starting the microprocessor program

The microprocessor program may be restarted from address zero by use of the 'R' command:

```
*;R
B8;ØØØØØØ
*
```

The microprocessor program counter is set to zero and the instruction register is cleared. All the hardware data-transfer flags are cleared and all peripherals are reset. The Peripheral Control Registers (Output Ports 1 and 2) are cleared, disabling all peripheral modes of operation, and the Data Display Port is cleared. The program WAITs at location zero.

15.2.5.2 Causing a program WAIT

A microprocessor WAIT may be occasioned at any time by use of the 'W' command. The current address at which the program waits is given, for example:

```
*;W
B8;ØØ1Ø13
*
```

15.2.5.3 Setting the Single-Step Mode

The microprocessor program may be run in the single-step mode of operation by means of the command ';nS' where n is any numeric

character in the range 1 to 7. For example:

```
*;3S
*
```

sets the single step mode. In this mode the program execution will proceed one machine cycle every time the proceed ('P') instruction is given (see Section 15.2.5.5).

15.2.5.4 Clearing the Single-Step Mode

The single step mode of operation may be cleared by use of the setting command with an argument of Ø or not specified. For example:

```
*;ØS          or          *;S
*                         *
```

both these commands disable the single-step mode of operation.

15.2.5.5 Proceed Instruction

The microprocessor program may be made to run by means of the proceed instruction:

```
*;P
```

Program execution will proceed for one machine cycle if the single-step mode has been set or until the next breakpoint is encountered if not. The running may be interrupted by means of the CTRL D command (section 15.1.1).

Optionally, a count may be specified. In this case program execution will proceed until the last-encountered breakpoint has been passed the specified number of times. For example

```
*;P                          (Proceed)
B3;ØØØ117                    (Breakpoint 3 encountered)
*1ØØ;P
```

causes the program to run until the 1ØØ th octal occurrence of breakpoint number 3. In this mode all other breakpoints are ignored although they remain set for future use.

15.2.6 Status and Data Examination

When running a program, DEBUG keeps a continual record of the microprocessor status and the contents of the data bus. These may be displayed on the teletype by use of the appropriate command.

15.2.6.1 Data Bus Contents

The contents of the microprocessor data bus during the current instruction may be examined by typing a 'D' followed by a slash character, for example:

```
*D/372
*
```

shows that the data bus contained the data 372_8. This must be interpreted with reference to the microprocessor status as described immediately below.

15.2.6.2 Status Examination

The microprocessor status during the current instruction may be examined by typing a 'S' followed by a slash character. The status will then be printed as an 8-bit value in octal form. The significance of each bit is given below. An additional facility is provided to aid interpretation - typing an 'X' character will cause the status byte to be expanded as an ASCII string using the mnemonics given below:

Bit	Mnemonic	Function
7	MEM RD	Indicates that the CPU is in the process of reading program or data

		from memory
6	OUT CY	Indicates that the CPU is in the process of transmitting data to an output port.
5	FETCH	Indicates that the CPU is in the FETCH cycle for the first byte of an instruction.
4	IN CYC	Indicates that the CPU is in the process of receiving data from an input port.
3	HALT	Indicates that program operation has ceased due to the execution of a HALT instruction.
2	STACK	Indicates that the CPU is in the process of performing a STACK operation.
1	MEM WRT	Indicates that the CPU is in the process of reading data from memory
Ø	INT CY	Indicates that the CPU has entered an interrupt cycle initiated by an external interrupt.

For example:

```
*S/24Ø X = MEM RD  FETCH
*
```

If the expansion (X) facility is not required then a carriage return character may be typed to obtain normal command mode.

15.2.7 Interrupts and Data Transfers

DEBUG contains the facility to simulate real-time interrupts from any peripheral including those which would occur in normal run-time operation from the PDP-11. It also has the facility to handle data transfers between the I8Ø8Ø and the PDP-11 in the manner of real-time transfers.

15.2.7.1 Interrupts

An interrupt may be transmitted to the microprocessor by means of the DEBUG 'T' facility. This must be followed by the octal byte which is to be the interrupt instruction, normally an 'RST' instruction. As the instruction is under user control, an instruction which is normally hard-wire associated with a particular peripheral may be simulated. For example:

```
*T/327 <CR>
*
```

sends the byte 327_8 to the CPU which is interpreted as an 'RST 3Q' instruction causing a transfer of program operation to the interrupt vector $3Ø_8$.

Note: The interrupt instruction may not be acknowledged immediately by the microprocessor and so DEBUG holds the relevant signals on the interface lines until this occurs. However, the interface lines are multiplexed and so the issue of a different mode of command (e.g. memory opening command requiring DMA mode) may cause spurious operation if given as the interrupt instruction is about to be read.

15.2.7.2 Data Transmission

Data may be output from the PDP-11 to the microprocessor by means of the 'O' command. For example

*0/224 <CR>

*

sends the data byte 224_8 to Input Port Ø of the microprocessor and sets the DATA flag (section 2.3.4). Precautions must be taken in issuing of command modes until this data has been read by the microprocessor (see Note in Section 15.2.7.1).

15.2.7.3 Data Reception

The mode of hardware operation is such that any data transmitted by the microprocessor is held on the interface lines until read by the PDP-11. Due to the multiplexed nature of the transmission lines this would effectively disable the normal operating modes of the DEBUG program. To overcome this difficulty DEBUG automatically reads and stores data transmitted to it. The program may be interrogated in respect of the data received by means of the 'I' command. DEBUG then types the last 16-bit word received (from Output Ports 377 and Ø - see Section 2.3.4) and the number of words received since it was last interrogated. For example:

*I/1Ø4326 (ØØØØØ3)

*

signifies that DEBUG has receivedthree words since the last interrogation and the last of these was $1Ø4326_8$. Both the buffer and the count are cleared when interrogated. If more than $65533_{1Ø}$ words are received between interrogations then the buffer will continue to be updated but the count will remain 'locked' at this value (177777_8) to avoid mis-interpretation by overflow to zero.

15.3 DEBUG CHARACTER SET AND COMMAND SUMMARY

15.3.1 Input Commands

The following characters are used as input commands to DEBUG

and in most cases are echoed identically on the teletype.

Ascii Code (octal)	Character	Function	Section
ØØ4	CTRL D	breaks an uncontrolled program loop	15.1.1
Ø12	LF	closes a memory location or I/O port and advances to the next	15.2.2.3 & and 15.2.3.3
Ø15	CR	closes a memory location or I/O port or a field terminator	
Ø57	/	field delimiter	
Ø6Ø to Ø67	Ø to 7	Octal digits used as address or data specifiers	15.2.1
Ø73	;	command delimiter	
Ø74	<	field delimiter specifying output port operations	15.2.3.2
Ø76	>	field delimiter specifying input port operations	15.2.3.1
1Ø2	B	Breakpoint specifier	15.2.4
1Ø4	D	Data bus specifier	15.2.6.1
111	I	Input data from micro-processor	15.2.7.3

117	O	Output data to microprocessor	15.2.7.2
12Ø	P	Proceed (run program)	15.2.5.5
122	R	Reset microprocessor	15.2.5.1
123	S	Status byte specifier or Single-step mode	15.2.6.1 or 15.2.5.3
124	T	Transmit an interrupt	15.2.7.1
127	W	Wait at current location	15.2.5.2
13Ø	X	eXpand status information	15.2.6.1
136	↑	closes a memory location or I/D port and opens the previous location or port (address -1)	15.2.2.3 and 15.2.3.3

15.3.2 Output Characters

In addition to the characters specified in Section 15.3.1, DEBUG may also output the following:

Ascii Code (octal)	Character	Function	Section
52	*	indicates readiness to accept another command	15.2
77	?	indicates receipt of an illegal	

		character (may only be illegal in context)	
136, 1Ø4	↑D	echoed in response to CTRL D	15.1.1

SECTION 16

'CALTST' TRANSDUCER CALIBRATION AND TEST PROGRAM

CALTST is a support program to the direct digital control facility to enable periodic calibration of the transducers of the system to be performed rapidly and efficiently. It also serves a test program to enable correct functioning of specific transducers to be checked.

16.1 CALLING AND USING 'CALTST'

CALTST normally resides on the system disk as a linked memory image module (CALTST.SAV). The first operation of CALTST when running is to load the MIcroprocessor Real-Time Operating System (MIRTOS.MLM) to the microprocessor so it is essential that this program is also resident on the system disk. CALTST is run using the standard monitor command:

```
.R CALTST
CALTST VØØ1-A
```

and then identifies itself as shown above.

Program flow then proceeds on a question/answer basis until the relevant operation is selected. This flow is basically self-explanatory and is detailed in Fig 16.1. It is necessary that the micro-machine system is running at normal synchronous speed as transducer operation is synchronised with shaft rotation by MIRTOS.

16.1.1 Gain Adjustment

To adjust the gain of any transducer it is necessary to run the relevant section of program as detailed in Fig 16.1. Once this is done, CALTST will respond by asking the true value (as indicated by a separate standard) of that parameter, which should be entered as a maximum of twelve figures including a decimal

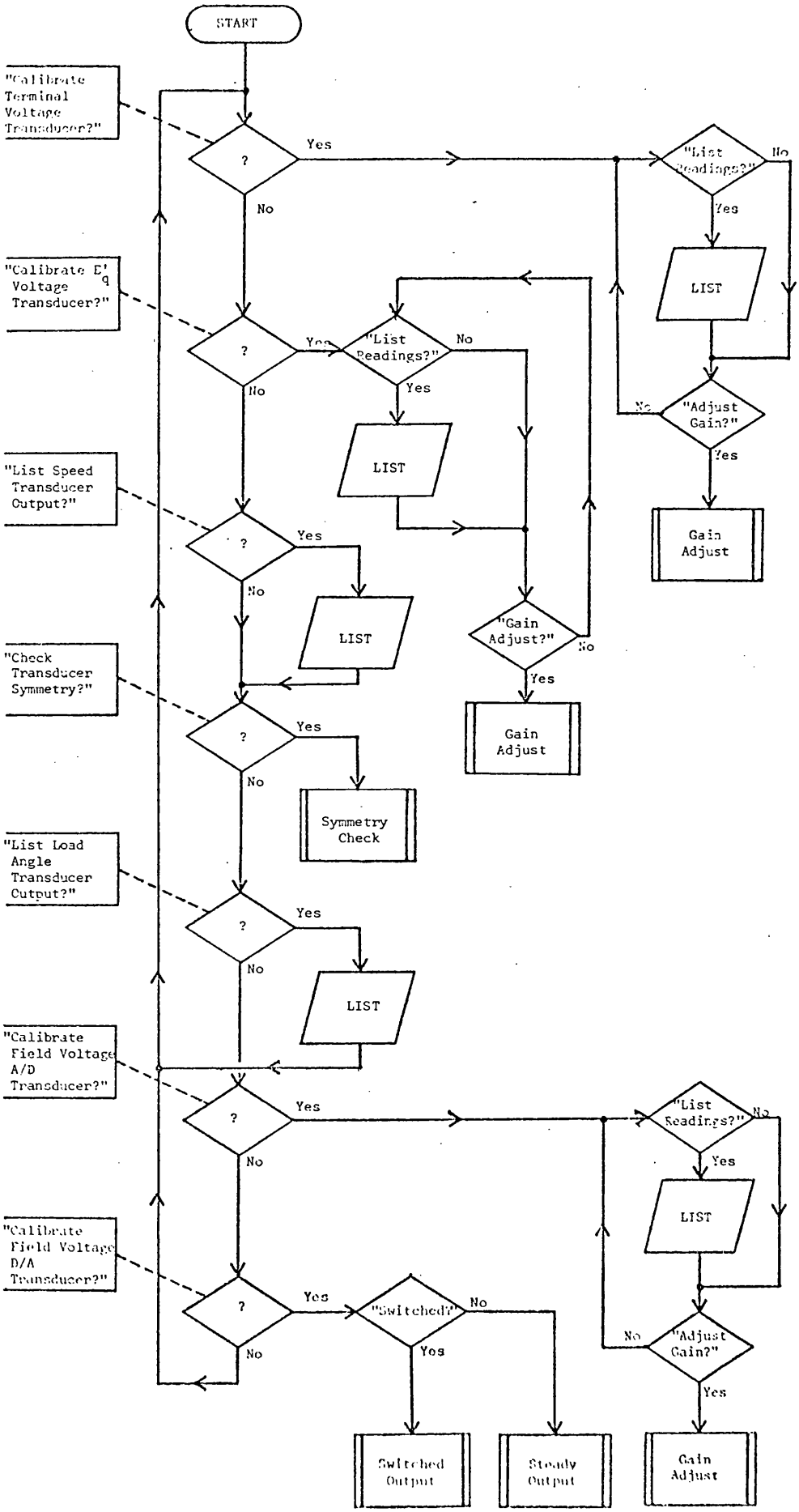


Fig. 16.1 'CALTST' Program Flow Diagram (Simplified)

point, for example:

ADJUST GAIN ? YES <CR>

TERMINAL VOLTAGE = 21Ø.Ø <CR>

CALTST will then display, on the microprocessor display port, the error difference between the true and indicated values in a manner described in section 17 (subroutine DISP). Hardware adjustment of the transducer may then be rapidly performed.

16.1.2 Output Listing

A listing of 25Ø (decimal) readings of any transducer may be performed by use of the appropriate commands with reference to Fig 16.1. CALTST will list 25Ø readings on the teletype followed by the mean and standard deviation of the readings. The converted value of these readings in the relevant parameters is also given.

16.1.3 Speed Transducer Symmetry

In the speed transducer symmetry check, the microprocessor display port displays the difference between the readings from transducers in opposite half-cycles of the shaft revolution. The transducer positions should be adjusted until this reading is a minimum as the display is one of magnitude only (unsigned).

16.2 ERROR MESSAGE SUMMARY

The following error messages may be output by CALTST during operation:

FETCH ERROR	An error has occurred in fetching a device handler. See IFETCH subroutine

CHANNEL ALLOCATION ERROR	An error has occurred during a channel allocation operation. See IGETC subroutine
LOOKUP ERROR	An error has occurred during a file lookup operation. See LOOKUP subroutine
ENTRY ERROR	An error has occurred during a file entry operation. See IENTER subroutine
ML LOADER ERROR (CODE = n)	An error has occurred during a microprocessor loading operation. See MPLOAD subroutine (section 17)
READ ERROR (CODE = n)	An error has occurred during a read (input from microprocessor) operation. See IREADW subroutine
WRITE ERROR (CODE = n)	An error has occurred during a write (output to microprocessor) operation. See IWRITW subroutine
MICROPROCESSOR ERROR (CODE = n)	An error has been detected in the operation of the microprocessor and its transducers. See MIRTOS program (section 18)

SECTION 17

'CONLIB' CONTROL SYSTEM LIBRARY

CONLIB is a library of subroutines, generally for use by Fortran programs, to provide some basic facilities for real-time direct-digital control systems. In addition to the on-line facilities it also contains useful support subroutines for off-line graph plotting and data manipulation.

17.1 CONTENTS

	Module	Entry/Csect	Entry/Csect
	LIST	LIST	
	DISP	DISP	
	IROLL	IROLL	
	IBIT	IBIT	
	SSWTCH	SSWTCH	
	MPSTRT	MPSTRT	
	MPDISP	MPDISP	
*	PLOTIN	PLOTIN	.$$$$.
*	ORIGIN	ORIGIN	
*	SCALE	SCALE	
*	POINT	POINT	
*	PLOT	PLOT	
*	LINE	LINE	
*	AXES	AXES	
	RANGE	RANGE	
	PLTDRV	GETPOS	MOVE
		PEN	PLOTER
		SCOPE	
	MPLOAD	MPLOAD	

17.2 USING 'CONLIB'

Any of the subroutines in CONLIB may be used by linking the CONLIB library with the object module of the user program in a LINK command [45], for example:

```
.R LINK
* prog = prog, sub, CONLIB/F <CR>
```

where 'prog' and 'sub' represent the user programs and subroutines respectively and /F is the default Fortran Library, if required. The subroutines within CONLIB comprise three major groups as described below.

17.2.1 General Purpose Routines

The general purpose routines (LIST, IROLL, IBIT and SSWTCH) may be used at any time within the PDP-11 system and do not require the presence of any additional hardware.

17.2.2 Microprocessor Control Subroutines

The microprocessor control subroutines (DISP, MPSTRT, MPDISP and MPLOAD) obviously provide functions which require that the PDP-11/20 computer is connected to an I8080 microprocessor system as described in other sections. In most cases, error messages will be given if this connection does not conform with the program requirements.

17.2.3 Plotter Subroutines

The plotter subroutines (PLOTIN, ORIGIN, SCALE, POINT, PLOT, LINE, AXES, RANGE and all subroutines in module PLTDRV) require that the PDP-11/20 is connected to a Bryan's X-Y plotter type 26000/A4, or equivalent, via the digital/analogue converter subsystem. (Connection details are given by the program 'SETUP' - section 18.4). In addition, those plot facility subroutines marked with an asterisk in section 17.1 require

an un-named common intercommunication block which must be reserved by the main program. This block is of size eight words and could be reserved by the following Fortran code:

```
DIMENSION IDUMMY (8)
COMMON IDUMMY
```

17.3 SUBROUTINE DETAILS

17.3.1 Subroutine LIST

Source Module: LIST.FOR

Object Library: CONLIB.OBJ

Fortran Call: CALL LIST (IVAR, SUM)

where: IVAR is the name of an integer variable array
SUM is a real variable into which will be returned the mean of the array IVAR

Description

A subroutine to list on the line printer the first 25Ø elements of the integer array IVAR in 25 rows of 1Ø elements. The subroutine also prints the mean value of the elements and the standard deviation expressed in both the units of the array and as a percentage.

Implementation

IVAR must be defined in the calling program as an integer array having at least 25Ø elements. The mean value of these elements is returned to the calling program as the argument of the real variable SUM.

17.3.2 Subroutine DISP

Source Module: DISP.FOR

Object Library: CONLIB.OBJ

Fortran Call: CALL DISP(DIFF)

where: DIFF is a real variable or real constant

Description

A subroutine to transmit an integer display to the microprocessor display port to give an indication during calibration routines. Generally, the display is used to indicate whether a transducer gain is 'high' or 'low' and the magnitude of the discrepancy.

Implementation

The real variable or constant DIFF is used to generate a transmitted value (I below) to be displayed in the microprocessor display port in the following form:

Value of DIFF	Transmitted value (I)	Display Port (No 3)
DIFF > Ø.Ø2	15	ØØØØ1111
Ø.Ø2 ≥ DIFF > Ø.Ø1	14	ØØØØ111Ø
Ø.Ø1 ≥ DIFF > Ø.ØØ5	12	ØØØØ11ØØ
Ø.ØØ5 ≥ DIFF > Ø.Ø	8	ØØØØ1ØØØ
DIFF = Ø.Ø	Ø	ØØØØØØØØ
Ø.Ø > DIFF > -Ø.ØØ5	16	ØØØ1ØØØØ
-Ø.ØØ5 ≥ DIFF > -Ø.Ø1	48	ØØ11ØØØØ
-Ø.Ø1 ≥ DIFF > -Ø.Ø2	112	Ø111ØØØØ
-Ø.Ø2 ≥ DIFF	24Ø	1111ØØØØ

Thus if DIFF is the fractional difference between an actual transducer reading and the expected (calibrated) reading, the bits in the display have the following significance:

Bit Ø	Reading > 2.Ø % high
Bit 1	Reading > 1.Ø % high
Bit 2	Reading > Ø.5 % high
Bit 3	Reading high
Bit 4	Reading low
Bit 5	Reading > Ø.5 % low
Bit 6	Reading > 1.Ø % low
Bit 7	Reading > 2.Ø % low

17.3.3 Subroutine IROLL

Source Module: IROLL.MAC

Object Library: CONLIB.OBJ

Fortran Call: CALL IROLL(IVAR)

where: IVAR is the name of an integer array to be 'rolled' (must be in INTEGER*2 format in consecutive memory locations).

Description

A subroutine to 'roll' the first 1ØØ elements of an array down 1Ø places in the array.

Implementation

This subroutine is primarily intended for use on an array IVAR (1Ø, LENGTH) to replace element IVAR(M,N) by element IVAR(M,N+1) for N less than 1Ø. That is, to roll the first 1Ø rows of the array forward.

The elements of IVAR must be in single-word format (INTEGER*2) and stored in consecutive memory locations.

17.3.4 Subroutine IBIT

Subroutine: IBIT

Source Module: IBIT.MAC

Object Module: CONLIB.OBJ

Fortran Call: I = IBIT (IDAT,MASK)

where: IDAT is an integer variable to be 'bit tested'.
MASK is an integer constant or variable to be used as the testing mask.

Description

A subroutine to test the bit pattern of the first argument using the second as a mask. This function is similar to the assembly language 'BIT' instruction.

Implementation

The value returned in I is the logical 'and' function of the values in IDAT and MASK. For example, if IDAT = 89 and MASK = 328 then I = 72:

```
IDAT  =  89  =  Ø ØØØ ØØØ ØØ1 Ø11 ØØ1   (binary)
MASK  = 328  =  Ø ØØØ ØØØ 1Ø1 ØØ1 ØØØ   (  "   )
                ---------------------
I     =  72  =  Ø ØØØ ØØØ ØØ1 ØØ1 ØØØ   (  "   )
```

17.3.5 Subroutine SWITCH

Source Module: SSWITCH.MAC

Object Library: CONLIB.OBJ

Fortran Call: CALL SSWITCH(I,J)

where: I is an integer or integer variable specifying the bit in the Console Switch Register to be tested

J is an integer variable into which the switch value is returned

Description

A subroutine to test specific bits in the Console Switch Register and return the values to the calling program.

Implementation

The value of I must be in the range Ø to $15_{1Ø}$ to check one of the sixteen bits, if out of this range then the value returned in J is 2. Otherwise, the value returned in J is as follows:

J contains a 1 if bit I is 1 (ON)
J contains a 2 if bit I is Ø (OFF)

17.3.6 Subroutine MPSTRT

Source Module: MPSTRT.MAC

Object Library: CONLIB.OBJ

Fortran Call: CALL MPSTRT

Description

A subroutine to reset the I8Ø8Ø microprocessor. All the peripheral transducers are also reset and all interrupt and auto modes of operation are disabled.

Implementation

The subroutine may be called at any time by a PDP-11 program to restart the microprocessor program. The microprocessor program counter and instruction register are set to zero and all peripheral interrupt and data transfer flags are reset. The peripheral interrupt and auto modes of operation are also disabled.

17.3.7 Subroutine MPDISP

Source Module: MPDISP.MAC

Object Library: CONLIB.OBJ

Fortran Call: CALL MPDISP(KODE)

where: KODE is an integer or byte constant or variable which will be displayed on the microprocessor display port.

Description

A subroutine to output an 8-bit byte to the microprocessor display port (Output Port No 3). Transfer is by direct memory access (HOLD) mode so there is no parity or other checking performed.

Implementation

If KODE is an integer constant or variable then only the low-order eight bits will be transmitted for display, no error is generated. Thus, normally, KODE will have a value in the range Ø to $225_{1Ø}$ to be of significance.

Example:

CALL MPDISP(1Ø7)

will cause the following pattern of lights to be displayed:

Ø11Ø1Ø11

17.3.8 Subroutine PLOTIN

Source Module: PLTSUB.FTN

Object Library: CONLIB.OBJ

Fortran Call: CALL PLOTIN

Description

A subroutine to initialise the internal plotter control variables and to ready the plotter for operation.

Implementation

The subroutine PLOTIN should be called as the first subroutine of the graphics package at execution time otherwise spurious results may occur. It may be called at any subsequent time to reset the plotter system.

The following operations are performed:

1. The pen is raised

2. The origin is set to the centre of the plotting field (location (Ø, Ø)).

3. The X and Y axis scales are set to 1ØØ.Ø user units per cm

4. The pen is moved to the origin (Ø, Ø).

17.3.9 Subroutine ORIGIN

Source Module: PLTSUB.FTN

Object Library: CONLIB.OBJ

Fortran Call: CALL ORIGIN (IX, IY)

where: IX is an integer constant or integer variable specifying the new location of the origin in the X axis

IY is an integer constant or integer variable specifying the new location of the origin in the Y axis.

Description

A subroutine to locate or shift the origin of a graph or other form of plotter output.

Implementation

The subroutine call shifts the origin to a point in the plotter field as specified by the coordinates IX and IY which should be in the following ranges:

$$-125Ø \leqslant IX \leqslant +125Ø$$
$$-9ØØ \leqslant IY \leqslant +9ØØ$$

The coordinates are in basic system units (refer to description of the subroutine MOVE) e.g. CALL ORIGIN (-125Ø, -9ØØ) locates the origin at the lower left hand corner of the plotter field.

NB. This subroutine provides for location of the origin at a basic level. If higher-level programs are required (suppressed-zero plots etc) then additional software must be provided.

17.3.10 Subroutine SCALE

Source Module: PLTSUB.FTN

Object Library: CONLIB.OBJ

Fortran Call: CALL SCALE (AX, AY)

where:

AX	is a real constant or real variable specifying the X axis scale in user units/cm
AY	is a real constant or real variable specifying the Y axis scale in user units/cm

Description

A subroutine to change the plotting scale of the X and/or Y axes.

Implementation

The X and Y axes scales are both initialised by the PLOTIN subroutine at 1ØØ.Ø user units/cm. A call to the subroutine SCALE will change both these scales to the values specified in the call. e.g.

CALL SCALE (5ØØ.Ø, Ø.1)

will set the X axis scale at 5ØØ.Ø user units/cm and the Y axis at Ø.1 user units/cm.

17.3.11 Subroutine POINT

Source Module: PLTSUB.FTN

Object Library: CONLIB.OBJ

Fortran Call: CALL POINT (X, Y, I)

where X is a real constant or real variable specifying the coordinate of the required location on the X axis.

Y is a real constant or real variable specifying the coordinate of the required location on the Y-axis.

Description

A subroutine to move the pen of the X-Y plotter to a specified location in the plotting field.

Implementation

The subroutine call moves the plotter pen to the coordinates (X,Y) where X and Y are specified in user units. A pen control operation is made prior to the move as determined by the value of the pen control variable I (refer to subroutine PEN for details). Thus, if the pen is currently at location (-1ØØ.Ø, 25.Ø) then

```
CALL POINT (2ØØ.Ø, 14Ø.Ø, 1)
```

will draw a continuous straight line to the coordinates (2ØØ.Ø, 14Ø.Ø).

NB. If the call is to a location outside the current legal plotter field spurious results may be obtained (refer to subroutine MOVE for details).

17.3.12 Subroutine PLOT

Source Module: PLTSUB.FTN

Object Library: CONLIB OBJ

Fortran Call: CALL PLOT (X, Y, NT)

where:

X	is a real constant or real variable specifying the coordinate of the required point on the X axis
Y	is a real constant or real variable specifying the cooordinate of the required point on the Y axis
NT	is an integer or integer variable specifying the type of character to be plotted

Description

A subroutine to plot a point in discrete point graph form.

Implementation

The subroutine call raises the plotter pen and moves it to the coordinates (X, Y) where X and Y are specified in user units. The character plotted is determined by the value of the integer NT:

Value of NT:	1	2	3	4	5
Character:	+	△	▽	◇	⅄

All characters are plotted Ø.2 cm high and are positioned centrally over the coordinates of the specified point.

NB. If the call is to a location outside the current legal plotter field spurious results may be obtained (refer to subroutine MOVE for details).

17.3.13 Subroutine LINE

Source Module: PLTSUB.FTN

Object Library: CONLIB.OBJ

Fortran Call: CALL LINE (X, Y, NL)

where:

- X is a real constant or real variable specifying the X axis coordinate of the point to which the line is to be drawn
- Y is a real constant or real variable specifying the Y axis coordinate of the point to which the line is to be drawn
- NL is an integer constant or integer variable specifying the form of the line

Description

A subroutine to draw a straight line from the present pen location to a new location specified by the user coordinates (X, Y).

Implementation

The type of line drawn is dependent on the value of the line control variable, NL, specified in the subroutine call:

Value of NL:	Line drawn:
1	Solid line
2	Broken line (Ø.2cm bar + Ø.2cm space)
3	Broken line (Ø.4cm bar + Ø.4cm space)
4	Broken line (Ø.8cm bar + Ø.8cm space)
5	Broken line (Ø.6cm bar + Ø.2cm space)

NB. If the call is to a location outside the current legal plotter field spurious results may be obtained (refer to subroutine MOVE for details)

17.3.14 Subroutine AXES

Source Module: PLTSUB.FTN

Object Library: CONLIB.OBJ

Fortran Call: CALL AXES

Description

A subroutine to draw a set of X and Y axes on a plotter field

Implementation

At the subroutine call the axes are drawn through the current location of the origin (as specified by a prior call to sub-

routine ORIGIN) to the limits of the plotter field in both directions. The axes are graduated at 1cm intervals.

17.3.15 Subroutine RANGE

Source Module: PLTSUB.FTN

Object Library: CONLIB.OBJ

Fortran Call: CALL RANGE (DMAX, DMIN, SIZE, BOT, TOP, UNIT)

where:

- DMAX is a real constant or real variable specifying the maximum value of a set of data points.
- DMIN is a real constant or real variable specifying the minimum value of a set of data points.
- SIZE is a real constant or real variable specifying the number of units into which the axis is to be divided.
- BOT is a real variable into which is returned the arithmetic minimum value of the axis side.
- TOP is a real variable into which is returned the arithmetic maximum value of the axis side.
- UNIT is a real variable into which is returned the value of one unit of the axis side.

Description

A subroutine to determine the optimum range for plotting a set of data points as graphical output.

Implementation

When called with the correct arguments, the subroutine determines the optimum range for plotting the data on a rough logarithmic series such that the scales will be typically Ø.1, Ø.2, Ø.5, 1.Ø, 2.Ø, 5.Ø etc user units/cm.

Further, if the range of data lies close to the origin (specifically |DMIN|/|DMAX| < Ø.5) then the zero is not suppressed and the zero point is included in the range.

A "STOP ØØØ1" command is given for irrecoverable errors, e.g. DMIN > DMAX in the subroutine call.

17.3.16 Subroutine GETPOS

Source Module: PLTDRV.MAC

Object Library: CONLIB.OBJ

Fortran Call: CALL GETPOS (NOWX, NOWY)

where: NOWX is an integer variable into which is returned the current value of the X coordinate

NOWY is an integer variable into which is returned the current value of the Y coordinate.

Description

A subroutine to return the current position of the plotter pen to the calling program.

Implementation

The subroutine returns the current position of the plotter pen as two coordinates in the range -2Ø48 to +2Ø47 (for significance of these coordinates see subroutine MOVE). The subroutine also re-aligns the stored memory position of the pen with its actual position. These may have been different at program

initialisation or if a DAC overflow condition has occurred.

17.3.17 Subroutine PEN

Source Module: PLTDRV.MAC

Object Library: CONLIB.OBJ

Fortran Call: CALL PEN(I)

where: I is an integer constant or integer variable controlling the pen raising and lowering function.

Execution time: 23.2 microseconds if no change in status of pen.
100 milliseconds is pen status change necessary.
32.8 microseconds if I ≠ Ø or 1.

Description

A subroutine to cause a raising or lowering of the ink pen on the X-Y plotter.

Implementation

The operation on the pen is determined by the value of the integer argument I:

Value of argument I	Operation
Ø	Raise pen (Not Write)
1	Lower pen (Write)
Other values	No change in pen position

A delay to allow the pen status operation to be performed is only implemented if an actual change in pen status is called. Thus repetitive calls with the same argument do not cause unnecessary delay in program operation.

17.3.18 Subroutine MOVE

Source Module: PLTDRV.MAC

Object Library: CONLIB.OBJ

Fortran Call: CALL MOVE(NX, NY)

where: NX is an integer constant or integer variable specifying an X coordinate.

NY is an integer constant or integer variable specifying a Y coordinate.

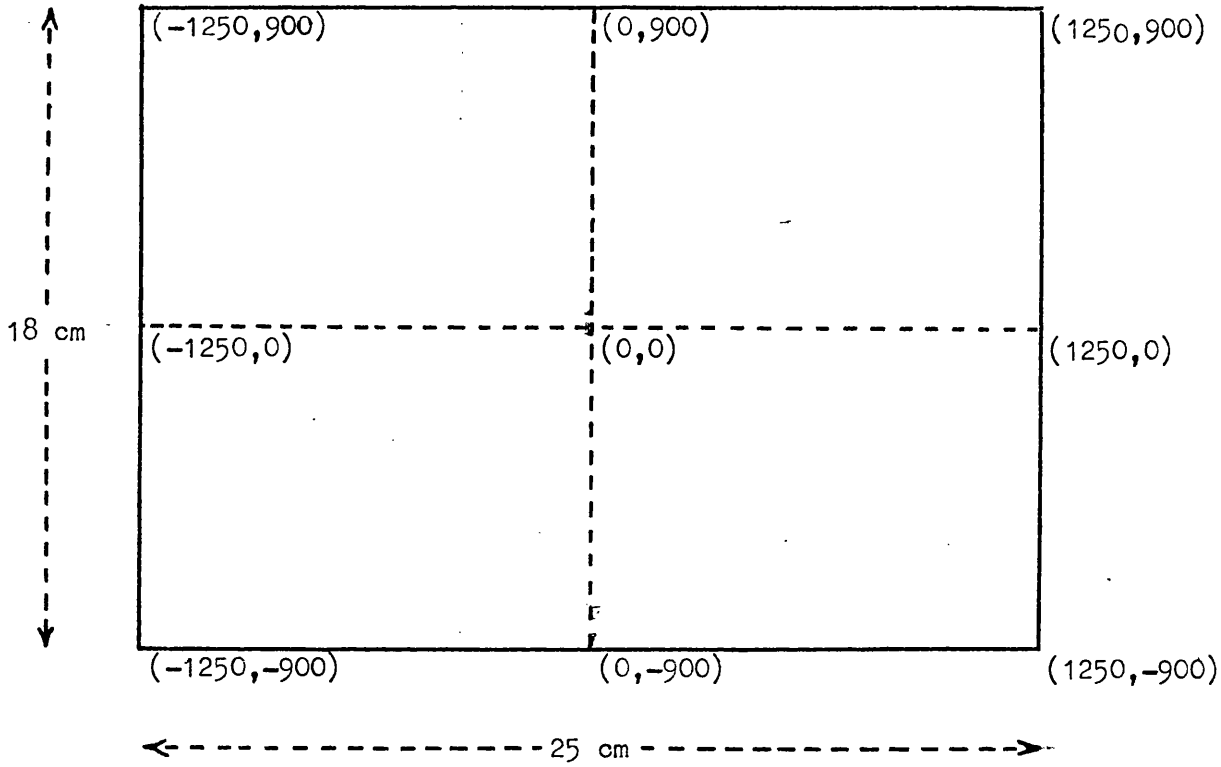
Execution time: 2ØØ mSec per cm of pen movement

Description

A subroutine to move the pen to a specified position on the X-Y plotter field. Pen movement is made in a straight line from the present pen position to the desired position with the pen in its current (raised or lowered) state. Movement is at a constant rate of 5.Ø cm/sec regardless of the direction of travel.

Implementation

The position to which the pen is moved is determined by the values of the integer arguments NX and NY. The physical plotter field is an area 25cm by 18 cm (A4 size) and this is divided into a grid of coordinates extending from -125Ø to +125Ø in the X axis and from -9ØØ to +9ØØ in the Y axis. Thus the basic plotter scale is 1ØØ units/cm in both axes (resolution Ø.Ø1 cm):



The plotter is driven by the D/A converter AA11-D and because of its output characteristics (p162 of the Peripherals and Interfacing Handbook) the following results will be obtained:

-125Ø < NX < +125Ø and -9ØØ < NY < +9ØØ

The pen will be moved to a position on the plotter field as described on the previous page.

-2Ø48 < NX < +2Ø47 and/or -2Ø48 < NY < +2Ø47 excluding the above region.

The pen will reach a limit stop on one of the axes of the X-Y plotter as it will be over-driven.

All other values of NX and NY

Spurious results will be obtained (eg 2Ø5Ø will be equivalent to -2, etc).

No errors will be reported.

17.3.19 Subroutine SCOPE

Source Module: PLTDRV.MAC

Object Library: CONLIB.OBJ

Fortran Call: CALL SCOPE

Description

A subroutine to facilitate the output of plot information on a graphics unit (oscilloscope) by eliminating the pen movement delays.

Implementation

Once called, the subroutine removes any delays incorporated in the pen raising and lowering and directional movement subroutines which are normally incorporated to allow for the plotter mechanical response time. Note that, if the plotter driver package is part of an overlay system then the overlaid version, when reobtained from disk, will have the delays incorporated.

17.3.20 Subroutine PLOTER

Source Module: PLTDRV.MAC

Object Library: CONLIB.OBJ

Fortran Call: CALL PLOTER

Description

A subroutine to restore the plotter subroutines to the mechanical plotter mode following a call to subroutine SCOPE.

Implementation

Once called, the subroutine restores all delays necessary to allow for the movement of the mechanical plotting system. See implementation notes for subroutine SCOPE with reference to overlaid systems.

17.3.21 Subroutine MPLOAD

Source Module: MPLOAD.MAC

Object Library: CONLIB.OBJ

Fortran Call: CALL MPLOAD ('DV:FILNAM.EXT')

Description

A subroutine to load a program to the I8Ø8Ø microprocessor using the ML device handler via the DR11-A interface and the microprocessor to PDP-11 Interface Card.

Implementation

The argument to the subroutine must be a valid filename specifier as described in Appendix O of the RT-11 System Reference Manual. If zero argument is used then the character string will be read from the system terminal. The following filename spcifiers are used:-

DV:	is the input device , default SY:
FILNAM	is the file name, no default assumed
.EXT	is the filename section, default .MLM

The following are valid calls:

```
CALL MPLOAD            Input name from teletype
I = MPLOAD (Ø)         Input name from teletype
CALL MPLOAD ('TEST') File DK:TEST.MLM loaded
CALL MPLOAD (ARRAY)  where ARRAY contains the literal filename
```

The subroutine call must precede any input/output operations as 'free' core is used dynamically for the load operation. In other cases the Fortran OTS buffer/handler integrity cannot be guaranteed.

The error values (I in the example I = MPLOAD (---)) are as follows:-

I = Ø	Normal return.
I = -1	Illegal command (bad separators, illegal file-name, command too long etc).
I = -2	A device specified is not found in the system tables (Note: Device ML: is always required).
I = -3	Insufficient free core for the buffer.
I = -4	An attempt to ENTER a file failed because of a full directory.
I = -5	An input file was not found in a LOOKUP.
I = -6	Switches detected in command string.
I = -7	An input or output error occurred during the load.

SECTION 18

REAL-TIME CONTROL PROGRAMS

This section mainly describes the operation of the real-time digital control program 'CONTRL' and its associated programs which run on the PDP-11/2Ø computer to produce the control functions described in the main text of this thesis. For its operation, it requires the presence of the Microprocessor Real-Time Operating System (MIRTOS) in the I8Ø8Ø microprocessor memory and the hardware configuration as described in Sections 1 to 11 of this Appendix. A brief description of the operation of the support programs SETUP, to calibrate the graph plotter facility, and MCPLOT, to plot the data stored by CONTRL, is also given.

18.1 GENERAL DESCRIPTION OF OPERATION

If the hardware of the real-time digital control system has not been in operation for some time, the calibration of the various analogue and analogue/digital conversion parts of the system may be checked by means of the programs CALTST (Section 16) and SETUP (Section 18.4). These programs are run under the modified RT-11 system monitor (Section 13).

Once satisfactory system calibration has been achieved, the control program may be run. The microprocessor operating system is loaded using the standard peripheral interchange program, PIP:

```
.R PIP
*ML:=MIRTOS.MLM
*↑C
.
```

The control program may then be run. As various time-out checks and low voltage checks are performed, it is desirable to synchronise the generator prior to running the control program to

avoid the occurrence of error conditions. In its present simplified form, the control program requests that specific values of the feedback state variable gains be entered, in order to test the effectiveness of these controls at any desired operating point of the synchronous generating system.

```
.R CONTRL
CONTRL VØ1-Ø1
K1(          ) K2(          ) K3(          )
```

The user may enter values for K_1 (e'_q), K_2 (δ) and K_3 ($p\delta$) which will then be employed by the computer in the feedback control.

Following a fault occurrence, the computer will ask whether the data collected as a result of the test is to be stored with the question:

```
STORE DATA ?
```

The reply is either yes (YE is sufficient) or no (NO or any other characters). If the reply is affirmative, the computer will then enquire whether a default scratch file (MCØØØØ.DAT) is to be used or a differently named file is to be generated. If a named file is to be generated, then this should be entered in response to the computer's asterisk. A typical example of this dialogue would be:

```
STORE DATA ?
YES
DEFAULT FILE (MCØØØØ.DAT) ?
NO
*MC2ØØ1.DAT
DATA STORED
```

Following the last message, the computer is again in control of the generating system ready for a new fault occurrence.

At any time, this sequence may be interrupted to plot the results

using the MCPLOT program:

```
↑C
↑C
.R MCPLOT
MCPLOT VØØ1-B
SWITCH INFORMATION ?
```

A complete description of the program operation is too lengthy to include here, due to the large number of plot permutations possible. However, the program operation is basically self-explanatory, and using the console switch controls and teletype commands it is possible to plot the major system states and output variables against time or each other on various scales. Auxilary facilities are provided for the suppression of axes, scale expansion, line printer listing and the super-position of plots.

18.2 'MIRTOS' MICROPROCESSOR REAL-TIME OPERATING SYSTEM

18.2.1 General Description

MIRTOS is the microprocessor real-time operating system which is used by the control program CONTRL and also by other auxilary programs, such as the calibration program CALTST. It contains the interrupt routines for the various transducers associated with the control system, as described in the section on hardware. It also contains the device handler subroutines for communication with the PDP-11 computer and various filtering and data processing routines, given in more detail in the program listing.

18.2.2 Run-time Messages

During the course of its operation, MIRTOS displays several numerical codes on the Data Display Port. These codes indicate the operating state of the program, as follows:

Code	Condition
000_8	Null. No valid program running
001_8	Waiting for load angle transducer signal in order to synchronise program operation with the machine shaft.
002_8	Waiting for digital filters to settle following a run-up operation. Feedback law is not valid at this time.
003_8	Normal run condition. Computer control of generator and feedback law valid.

18.2.3 Error Codes

If MIRTOS detects a system malfunction, it attempts to override the fault and maintain control of the generating system. This is not possible in all cases, depending on the seriousness of the error, and a system crash may result. A complete description is given in the program listing. MIRTOS attempts to diagnose the error to the best of its ability and display an error code on its Display Port. However, it should be noted that in the case of certain complex multiple-error conditions misleading error codes may arise. The following error codes may occur:

Code	Error Condition
201_8	Error in data from PDP-11/20. A parity error has been detected in the data from the PDP-11, field voltage will be restored to its steady-state value.
205_8	Unused interrupt routine entered. The program has reached location 50_8 either as a result of an illegal program jump or a spurious external interrupt.

206_8	Similar to error above, at location 60_8.
207_8	Similar to error above, at location 70_8.
210_8	Insufficient data. Too few bytes in transmitted data block (compared to byte count).
211_8	Excess data. Too many bytes in transmitted data block (compared to byte count).
212_8	DELTA1 not done. Load angle transducer sampling not completed in its allotted time interval.
213_8	DELTA2 not done. As error above.
214_8	VF A/D not done. Field voltage conversion not completed in allotted time interval.
215_8	VT A/D not done. Terminal voltage conversion not completed in allotted time interval.
216_8	EQ A/D not done. E'_q voltage conversion not completed in allotted time interval.
220_8	Time out on data from PDP-11. A response has not been obtained from the PDP-11 in the allotted time interval.
221_8	Time out on data to PDP-11. Previous data transmission not completed at start of new data transmission.

18.3 'MCPLOT' GRAPH PLOTTING PROGRAM

MCPLOT is a moderately sophisticated graph plotting program and a

complete description of all its operating modes would be too long for inclusion here. However, it is basically self explanatory in operation, as simply illustrated by means of the following example. This example plots the field voltage curves for three tests on a single sheet of paper. Two sets of results are stored on magnetic tape and one is on disk.

```
.R MCPLOT
MCPLOT VØØ1-B
SWITCH INFORMATION ?
YES
SWITCH:          ACTION:
  Ø              CHANGE FILE NAME SPEC
  1              PLOT FIELD VOLTAGE
  2              PLOT TERMINAL VOLTAGE
  3              PLOT EQ'
  4              PLOT DEL EQ'
  5              PLOT DELTA
  6              PLOT DEL DELTA
  7              PLOT OMEGA
  8              PLOT DEL OMEGA
  9              PLOT FREE-TIME CLOCK
 1Ø              PLOT FEEDBACK CONTROL
 11              PHASE PLANE PLOT
 12              SUPPRESS AXIS
 13              SUPPRESS LINE PRINTER
 14              WAIT FOR NEXT PLOT
 15              CHANGE SCALE ETC
PAUSE ---- SET SWITCHES
```

The program pauses at this point to await the switch action. It is convenient that the operator loads the plotter with paper during this interval. The operator also sets switches 0,1 and 13 on and types a carriage return character. The program then requests the file names:

```
FILE NAMES ?
```

```
MT:MCØØ47.DAT          (The operator enters the file names.
MT:MC1ØØ1.DAT           The default device is SY:.  The
MCØØ33.DAT              sequence is terminated by a blank
                        line)
```

The program now proceeds to plot the requested curves. Once the axes have been drawn for the first curve, switch 12 may be set on to eliminate repeat drawings of the axes.

18.4 'SETUP' PLOTTER CALIBRATION ROUTINE

SETUP is a PDP-11 mainline program to facilitate setting and calibration of the Bryan's 26ØØØ X-Y plotter used with the direct digital control system. It is run using a standard form of command:

```
.R SETUP
SETUP VØ2-Ø1
NEED ANY HELP ?
```

In reply to this question, the user may anser yes (YE) or no (any other characters). If the answer is affirmative, the program will proceed in a fully self-explanatory manner to guide the user in the correct settings of the plotter. If negative, only code numbers relevant to the setting operations will be output. This latter mode enables rapid calibration to be performed by experienced users. These operations are briefly summarised as:

Code	Operation
ØØØØØØ	Set pen in the centre (zero) of the plotting field using the 'set zero' controls.
ØØØØØ1	Set pen in lower left hand corner of plotting field using 'cal' controls.

ØØØØØ2	Check that pen is in upper right hand corner of plotting field.

The figure plotted should measure 18 by 25 cms.

REFERENCES

1. BRUCK R.W. and MESSERLE H.K.: "The capability of alternators", Proc IEE, 1955, Vol 102(A), pp 611-618.
2. BOLLINGER K. and LALONDE R.: "Tuning synchronous generator regulators using on-line generator models", IEEE Trans, 1977, Vol PAS-96, pp 32-37.
3. HUGHES F.M.: "Improvement of turbogenerator transient performance by control means", Proc IEE, 1973, Vol 120, pp 233-240.
4. SULLIVAN A.C. and LEE H.: "Fast governing of turbogenerators during transients", Ibid, pp 371-378.
5. CUSHING E.W., DRECHSLER G.E., KILLGOAR W.P., MARSHALL H.G. and STEWART H.R.: "Fast valving as an aid to power system transient stability and prompt resynchronisation and rapid reload after full load rejection", Paper 71TP7105-PWR presented at 1971 IEEE-ASME Power Generation Conference, St Louis, Mo, Sept 1971.
6. AANSTAD O.J. and LOKAY H.E.: "Fast valve control can improve generator response to transient disturbances", Westinghouse Engineer, July 1970.
7. WILSON A.T. and PORAY A.T.: "Comparison of methods of improving the transient performance of power stations", System Technical Planning Report No PL-ST/3/73, CEGB.
8. O'KELLY D. and MUSGRAVE G.: "Improvement of power system transient stability by phase-shift insertion", Proc IEE, 1973, Vol 120, pp 247-252.

9. REITAN D.K. and RAO N.R.: "A method of improving transient stability by bang-bang control of tie-line reactance", IEEE Trans, 1974, Vol PAS-93, pp 303-311.

10. DINELEY J.L., MORRIS A.J. and PREECE C.: "Optimised transient stability from excitation control of synchronous generators", Ibid, 1968, Vol PAS-87, pp 1696-1705.

11. JACOVIDES L.J. and ADKINS B.: "Effect of excitation regulation on synchronous machine stability", Proc IEE, 1966, Vol 113, pp 1021-1034.

12. DINELEY J.L. and MIKHAIL S.E.: "The dynamic stability of a synchronous generator under leading power factor conditions", IEEE Trans, 1975, Vol PAS-94, pp 288-298.

13. BERG R.O.: "LSI and the equipment designer", Ibid, 1971, Vol IGA-7, pp 504-509.

14. COHN N, BIDDLE S.R., LEX R.G., PRESTON E.H., ROSS C.W. and WHITTEN D.R.: "On-line computer applications in the electric power industry", Proc IEEE, 1970, Vol 58, pp 78-87.

15. JERVIS M.W.: "On-line computer in power stations", Proc IEE, 1972, Vol 119, pp 1052-1076.

16. DANIELS A.R., LEE Y.B. and PAL M.K.: "Non-linear power system optimisation using dynamic sensitivity analysis", Ibid, 1976, Vol 123, pp 365-370.

17. SPEEDY C.B., BROWN R.F. and GOODWIN G.C.: "Control theory: Identification and optimal control", Oliver and Boyd, Edinburgh, 1970.

18. DEMELLO F.P. and CONCORDIA C.: "Concepts of synchronous machine stability as affected by excitation control", Trans IEEE, 1969, Vol PAS-88, pp 316-329.

19. DANDENO P.L., KARAS A.N., McCLYMONT K.R. and WATSON W.: "Effect of high-speed rectifier excitation systems on generator stability limits", Ibid, 1968, Vol PAS-87, pp 190-201.

20. YU Y.N., VONSURIYA K. and WEDMAN L.N.: "Application of an optimal control theory to a power system", Ibid, 1970, Vol PAS-89, pp 55-62.

21. RAMAMOORTY M. and ARMUGAM M.: "Design of optimal regulators for synchronous machines", Ibid, 1973, Vol PAS-92, 269-277.

22. RAMAN S. and KAPOOR S.C.: "Synthesis of optimal regulator for a synchronous machine", Proc IEE, 1972, Vol 119, pp 1383-1390.

23. ANDERSON J.H.: "The control of a synchronous machine using optimal control theory", Proc IEEE, 1971, Vol 59, pp 25-35.

24. NEWTON M.E. and HOGG B.W.: "Optimal control of a micro-alternator system", Trans IEEE, 1976, Vol PAS-95, pp 1822-1833.

25. BARTLETT J.P., GIBBARD M.J. and WOODWARD J.L.: "Performance of a 5 KVA synchronous generator with an optimal excitation regulator", Proc IEE, 1973, Vol 120, pp 1250-1256.

26. DAVISON E.J. and RAU N.S.: "The optimal output feedback control of a synchronous machine", Trans IEEE, 1971, Vol PAS-90, pp 2123-2134.

27. RAMAMOORTY M. and ARUMUGAN M.: "Design of optimal constant-output feedback controllers for a synchronous machine", Proc IEE, 1972, Vol 119, pp 257-260.

28. ARUMUGAM M. and RAMAMOORTY M.: "A dynamic observer for a synchronous machine", Int J Control, 1972, Vol 15, pp 1129-1136.

29. ELMETWALLY M.M., RAO N.D. and MALIK O.P.: "Experimental results on the implementation of an optimal control for synchronous machines", Trans IEEE, 1975, Vol PAS-94, pp 1192-1200.

30. MUKHOPADHYAY B.K. and MALIK O.P.: "synchronous machine optimisation with modified quasilinearisation technique", Ibid, 1972, Vol PAS-91, pp 1-9.

31. IYER S.N. and CORY B.J.: "Optimisation of turbo-generator transient performance by differential dynamic programming", Ibid, 1971, Vol PAS-90, pp 2149-2157.

32. LEE Y.B.: "Sensitivity and optimal control studies of power systems", PhD Thesis, University of Bath, 1975.

33. GILL P.E. and MURRAY W.: "Quasi-Newton methods for unconstrained optimisation", J Inst Maths Applics, 1972, Vol 9, pp 91-108.

34. MARTIN M.A.: "Micromachine studies of power systems", MSc Thesis, University of Bath, 1973.

35. ADKINS B.: "Micromachine studies at Imperial College", Electrical Times, July 7th, 1960, p3.

36. HAMMONS T.J. and PARSONS A.J.: "Design of a micro-alternator for power system stability investigations", Proc IEE, 1971, Vol 118, pp 1421-1441.

37. ADKINS B. and WIDGER G.F.T.: "Progress with micro-machine research at Imperial College", Electrical Times, Aug 6th, 1971, pp 29-33, Aug 20th, 1971, pp 39-41.

38. HAMMONS T.J., HOGG B.W. and LEWIS D.G.: "A micro-alternator system for power system stability studies", Proc 7th Univ Power Eng Conf, Bradford, 1972, pp 209-221.

39. HUBER D.W., HOPE G.S. and MALIK O.P.: "Synchronous machine field time-constant regulator", Proc IEEE, 1973, Vol 61, pp 1152-1153.

40. HARLEY R.G. and ODENDAL E.J.: "Regulating the field resistance of a microalternator", Proc IEE, 1975, Vol 122, p 278.

41. PARK R.H.: "Definition of an ideal synchronous machine and formula for the armature flux linkages", Gen Elect Rev, 1928, Vol 31, pp 332ff.

42. PARK R.H.: "Two-reaction theory of synchronous machines: Generalised methods of analysis - Part 1", Trans Amer Inst Elect Engmrs, 1929, Vol 48, pp716ff, ditto, Part 2, 1933, Vol 52, pp 353ff.

43. PAL M.K.: "Mathematical methods in power systam stability studies", PhD Thesis 1971, University of Aston in Birmingham.

44. CHANA G.S.: "Optimal Control Studies of Power System

Stability", PhD Thesis, (To be published), University of Bath.

45. "RT-11 system reference manual", Digital Equipment Corporation, Publication No DEC-11-ORUGA-C-D, 1976.

46. "Processor handbook, PDP-11/20, -11/15, -11/R20", Digital Equipment Corporation, 1972.

47. DANIELS A.R. and PAL M.K.: "The effects of excitation control of turboalternators on power system stability and performance", Final report on CEGB research contract, 1974, University of Bath.

48. "RT-11 software support manual", Digital Equipment Corporation, Publication No DEC-11-ORPGA-B-D, 1976.

49. EXON J.L.T.: "Variable characteristic synchronous machine", PhD Thesis, University of Bath, 1970.

50. MOYA O.E.O. and CORY B.J.: "Online control of generator transient stability by minicomputer", Proc IEE, 1977, Vol 124, pp 252-258.

51. BURROWS P.J.: "Investigations into direct digital control systems", Lucas Aerospace Ltd, Progress Report, April 1975.

52. BOGNOR I. and KAZDA L.F.: "An investigation of the switching criteria of high order contactor servo-mechanisms", Trans AIEE, 1954, Vol 73, pp 118-127.

53. TOU J.T.: "Modern control theory", McGraw-Hill, New York, 1964.

54. COALES J.F. and NOTON A.R.M.: "An on-off servo-mechanism with predicted change-over", Proc IEE, Vol 103, pp 449-462.

55. ATHANASSIADES M. and SMITH O.J.M.: "Theory and design of high-order bang-bang control systems", IRE Trans Aut Control, 1961, Vol AC6, pp 125-134.

56. CLEMENT P.R.: "On the bang-bang regulator problem for linear sampled-data systems", IEEE Trans, 1963, Vol AC-6, pp 180-182.

57. RAJAGOPALAN A. and HARIHARAN M.V.: "Bang-bang excitation control", IEEE Trans, 1974, Vol PAS-93, pp 703-711.

58. JONES G.A.: "Transient stability of a synchronous generator under conditions of bang-bang excitation scheduling", Ibid, 1965, Vol PAS-84, pp 114-121.

59. WALTZ M.D. and FU K.S.: "A heuristic approach to reinforcement learning control systems", IEEE Trans Aut Control, Vol AC-10, 1965, pp 390-398.

60. JONES L.E.: "On the choice of sub-goals for learning control systems", Ibid, 1968, Vol AC-13, pp 613-621.

61. ASHER R.B., ANDRISANI D. and DORATO P.: "Bibliography on adaptive control systems", Proc IEEE, 1976, Vol 64, pp 1226-1240.

62. CONCORDIA C.: "Effect of prime-mover speed control characteristics on electric power system performance", IEEE Trans, 1969, Vol PAS-88, pp 752-756.

63. COLES H.E.: "Effects of prime-mover governing and voltage regulation on turbo-alternator performance", Proc IEE, 1965, Vol 112, pp 1395-1405.

64. COLES H.E.: "Dynamic performance of a turboalternator utilising 3-term governor control and voltage regulation", Ibid, 1968, Vol 115, pp 266-279.

65. DINELEY J.L. and KENNEDY M.W.: "Influence of governors on power-system transient stability", Ibid, 1964, Vol 111, pp 98-106.

66. DANIELS A.R., LEE Y.B. and PAL M.K.: "Combined suboptimal excitation control and governing of a.c. turbogenerators using dynamic sensitivity analysis", Ibid, 1977, Vol 124, pp 473-478.

67. "Intellec 8/Mod 80 Microcomputer Development System Reference Manual", Intel Corporation, Publication Number MCS-307-1074/3K.

68. "DR11-C General Device Interface Manual", Digital Equipment Corporation, Publication No EK-DR11C-TM-004, 1974.

69. "PDP-11 Peripherals and Interfacing Handbook", Digital Equipment Corporation 1971.

70. "PDP-11 Paper Tape Software Programming Handbook", Digital Equipment Corporation, Publication No DEC-11-GGPC-D, 1972.

71. "RT-11 V02C System release notes", Digital Equipment Corporation, Publication No DEC-11-ORNRA-A-D, 1976.

72. "INTEL 8080 Assembly Language Programming Manual", Intel Corporation, Publication No 26-277-8K, 1974.